

IMT REPORTING SYSTEM STORED PROCEDURES (USER)

STORED PROCEDURE
Add keys
Addrptdef
Autosetup wizard
Cascade update
Concat c codes
Concat p codes
Create r101 topline
Del key
Do101 flow seldatfix
Doavail401keys
Doavail401keysbyid
Doavailkeys
Doavailkeysbyid
Dobaserankinfo
Docampesets
Docamprsets
Doestmaxeffort
Domw
Dopanelesets
Dopanelrsets
Doprodsetup
Dor101 summary
Dor401 summary
Dorecyclekey
Dorecyclepanelkey
Dosourceinds
Dosummary
Dosummarydb
Expenses
Getcorpinfo
Maxbilleffort
Maxeffort
Maxgifteffort
Pagedquery
Rebuild rpt cats
Show columns
Sp add autosetup schemes
Sp add campaign
Sp add campaign def
Sp add corp panel sub type
Sp add corp panel test type
Sp add corp panel type
Sp add keys
Sp add keys2
Sp add panel
Sp add panel def
Sp add prior source desc
Sp add prod camp status

STORED PROCEDURE
Sp add prod camp type
Sp add prod e set
Sp add prod list cat
Sp add prod list name
Sp add prod list segment
Sp add prod panel package
Sp prod r set
Sp add renewal psdef
Sp add source
Sp add source description
Sp cascade product id
Sp deletecamp statuses
Sp deletecamp types
Sp deletepanel subtypes
Sp deletepanel testtypes
Sp deletepanel types
Sp deleterpt def
Sp edit autosetup
Sp edit autosetup campaign def
Sp edit autosetup panel def
Sp edit autosetup renewal psdef
Sp edit campaign
Sp edit corp panel sub type
Sp edit corp panel test type
Sp edit corp panel type
Sp edit key
Sp edit panel
Sp edit prior source desc
Sp edit prod camp status
Sp edit prod camp type
Sp edit prod e set
Sp edit prod list cat
Sp edit prod list name
Sp edit prod list segment
Sp edit prod panel package
Sp edit prod r set
Sp edit source
Sp edit source description
Sp loginfo
Sp deletecamp types
Sp deletepanel subtypes
Sp deletepanel testtypes
Sp deletepanel types
Sp deleterpt def
Sp edit autosetup
Sp edit autosetup campaign def
Sp edit autosetup panel def
Sp edit autosetup renewal psdef
Sp edit campaign
Sp edit corp panel sub type
Sp edit corp panel test type
Sp edit corp panel type

STORED PROCEDURE
Sp edit key
Sp edit panel
Sp edit prior source desc
Sp edit prod camp status
Sp edit prod camp type
Sp edit prod e set
Sp edit prod list cat
Sp edit prod list name
Sp edit prod list segment
Sp edit prod panel package
Sp edit prod r set
Sp edit edit source
Sp edit source description
Sp loginfo
Sp removeduplicate101keys
Sp removeduplicate401keys
Sp updatecamp statuses
Sp updatecamp types
Sp updatepanel subtypes
Sp updatepanel testtypes
Sp updatepanel type
Sp getrptlist

Procedure Name: Add_keys

```
/***** Object: Stored Procedure dbo.add_Keys Script Date: 1/14/99 3:43:07 PM *****/  
/***** Object: Stored Procedure dbo.add_Keys Script Date: 1/12/99 11:05:37 PM *****/  
/***** Object: Stored Procedure dbo.add_Keys Script Date: 11/28/98 2:31:13 PM *****/  
/***** Object: Stored Procedure dbo.add_Keys Script Date: 11/17/98 2:43:36 PM *****/  
/***** Object: Stored Procedure dbo.add_Keys Script Date: 10/19/98 6:03:58 PM *****/  
/***** Object: Stored Procedure dbo.add_Keys Script Date: 10/6/98 7:56:48 PM *****/  
/***** Object: Stored Procedure dbo.add_Keys Script Date: 10/6/98 11:08:02 AM *****/
```

```
@full_promo_key varchar (25),  
@dt_valid_from smalldatetime,  
@dt_valid_to smalldatetime,  
@src_cat_name varchar(30),  
@src_indicator char (3),  
@panel_name varchar (250),  
@camp_name varchar (250),  
@product_id numeric(8,0),  
@source_name varchar(250)
```

AS

insert into keys

```
(  
    full_promo_key,  
  
    dt_valid_from,  
    dt_valid_to,  
    src_cat_name,  
    src_indicator,  
    panel_name,  
    camp_name,  
    product_id,  
    source_name  
)  
values  
(  
    @full_promo_key,  
    @dt_valid_from,  
    @dt_valid_to,  
    @src_cat_name,  
    @src_indicator,  
    @panel_name,  
    @camp_name,  
    @product_id,  
    @source_name  
)
```

Delete from avail_keys where full_promo_key = @full_promo_key and product_id=@product_id



Procedure Name: Addrptdef

```
/*
    (
        @parameter1 datatype = default value,
        @parameter2 datatype OUTPUT
    )
*/
(@rpt_group varchar(75),
 @rpt_title varchar(75),
 @rpt_description varchar(250),
 @rpt_selFormula varchar(1000),
 @rpt_path varchar(250),
 @rpt_param1 varchar(50),
 @rpt_param2 varchar(50),
 @rpt_param3 varchar(50),
 @rpt_param4 varchar(50),
 @product_id varchar(250),
 @login varchar(50)
)

As
    /* set nocount on */
    Insert into rpt_Defs
    (rpt_group,rpt_title,rpt_description,rpt_selFormula,rpt_path,rpt_param1,rpt_param2,product_id)
    Values
    (@rpt_group,@rpt_title,@rpt_description,@rpt_selFormula,@rpt_path,@rpt_param1,@rpt_param2,@product_id)

execute sp_LogInfo @product_id,@login,'Saved Report',@rpt_title
```

**Procedure Name: Autosetupwizard**

@verbose int -- 0 means no text, non-zero means display text messages
, @product_id int

as
begin --autosetup wizard
if @product_id > 0 --product id is set to -1 to run autosetup_wizard for ALL PRODUCTS. This happens is
doavail401keys
begin

```
    declare avail_keys cursor
    for select product_id
        , full_promo_key
        , fulfillment_house
        , dt_valid_to
        , source_identifier
        , mail_qty
    from avail_keys
    where fulfillment_house is not null
    and product_id = @product_id
    --or product_id = 360
    --or full_promo_key like "6RAA_K_529"
```

end
else
begin

```
    declare avail_keys cursor
    for select product_id
        , full_promo_key
        , fulfillment_house
        , dt_valid_to
        , source_identifier
        , mail_qty
    from avail_keys
    where fulfillment_house is not null
```

end

--Avail Key Vars

```
--declare @product_id varchar(50)
declare @avail_key varchar(50)
declare @fulfillment_house varchar(50)
declare @dt_valid_to datetime
declare @source_identifier char(1)
declare @mail_qty int
```

--Campaign vars

```
declare @campaign_code varchar(50)
declare @campaign_description varchar(50)
declare @rec_id int
declare @src_indicator char(3)
declare @camp_name varchar(250)
declare @title_name varchar(250)
declare @camp_date_fr datetime
```

--Setup Wizard vars

```

declare @setup_mask varchar(50)
declare @src_cat_name varchar(50)
declare @source_name varchar(250)
declare @prior_source_indicator varchar(2)
declare @valid_from_date datetime
declare @valid_to_date datetime
declare @grp_name varchar(50)
declare @e_set_name varchar(50)

--panel vars
declare @panel_name varchar(250)
declare @panel_code varchar(250)
declare @panel_scheme varchar(50)

--local vars
declare @i int
declare @number_of_valid_schemes int

if @verbose <> 0 print "Starting ..."
open avail_keys
fetch next from avail_keys into @product_id, @avail_key, @fulfillment_house, @dt_valid_to,
@source_identifier, @mail_qty
while @@fetch_status = 0 --Loop through avail keys and attempt to assign them to campaigns
begin
    if @verbose <> 0 print "Avail Key " + @avail_key
    if @verbose <> 0 print "Source Identifier " + @source_identifier
    --Get autoseup vars for schemes matching current key
    set @setup_mask = null
    --If the number of valid schemes is greater than zero and the source category is renewals
    then the source is split
    --and a match is required from the prior source definition table(autoseup_renewal_psdef)
    select @number_of_valid_schemes = count(*)
        from autoseup_scheme
        where product_id = @product_id
        and valid_from_date <= @dt_valid_to
        and valid_to_date >= @dt_valid_to
        and src_indicator = @source_identifier
    --GREGG ADDED THE FOLLOWING CONDITION TO THE VALID
    and setup_mask = @setup_mask
    select @setup_mask = setup_mask
        , @src_cat_name = src_cat_name
        , @src_indicator = src_indicator
        , @source_name = source_name
        , @valid_from_date = valid_from_date
        , @valid_to_date = valid_to_date
        , @grp_name = grp_name
        , @e_set_name = isnull(e_set_name, "Undefined")
        from autoseup_scheme
        where product_id = @product_id
        and valid_from_date <= @dt_valid_to
        and valid_to_date >= @dt_valid_to
        and src_indicator = @source_identifier
        and len(setup_mask) = len(@avail_key)

```

```

if @verbose <> 0 print "Source Indicator ==> " + @src_indicator
if @setup_mask is not null --Build campaign code based on setup mask, otherwise move
on to next scheme
begin
    set @campaign_code = null
    set @camp_name = null

    --if len(@avail_key) = 11 and @src_cat_name = "RENEWALS" set
    @setup_mask = "SxxxxZZxCCC"
    if @verbose <> 0 print "Found an applicable setup mask! ==> " + @setup_mask
    if @fulfillment_house = "CENTROBE" or @fulfillment_house = "PALM
COAST" exec concat_c_codes @setup_mask, @avail_key, @campaign_code output
    if @fulfillment_house = "CDS" and (@src_cat_name = "DTP" or
@src_cat_name = "GIFT") exec concat_c_codes @setup_mask, @avail_key, @campaign_code output
    if @fulfillment_house = "CDS" and (@src_cat_name = "RENEWALS" or
@src_cat_name = "BILLING")
    begin --set code to pos2 & last pos then match value with def table and set
description
        set @campaign_code = substring(@avail_key,2,1) +
substring(@avail_key,len(@avail_key),1)

    end --if
    if @verbose <> 0 print "campaign code ==>" + @campaign_code

    select @camp_name = campaign_description
        , @camp_date_fr = campaign_date
        from autosetup_campaign_def
    where campaign_code = @campaign_code
    --and source_identifier = substring(@avail_key,1,1)
        and src_cat_name = @src_cat_name
        and product_id = @product_id

    if @camp_name is null --Build new campaign name
    begin
        set @camp_name = ""
        if @verbose <> 0 print "Source Category Name ==> " +
@src_cat_name

        if @verbose <> 0 print "Fulfillment House ==> " +
@fulfillment_house

        if @fulfillment_house = "CENTROBE"
        begin
            if @src_cat_name = "DTP" or @src_cat_name = "GIFT" set
            @camp_name = substring(@avail_key, 6, 4) + " Campaign " + @campaign_code
            --if @src_cat_name = "RENEWALS" set @camp_name =
            "Expire " + @campaign_code

            if @src_cat_name = "RENEWALS"
            begin
                set @camp_name = null
                select @camp_name = campaign_description
                    , @camp_date_fr = campaign_date

                    from autosetup_campaign_def
                    where src_cat_name = @src_cat_name
                    and campaign_code = @campaign_code

```



```

                                and product_id = @product_id
                                if @camp_name is null
                                begin
                                    set @camp_name = "Expire " +
@campaign_code

                                    set @camp_date_fr = null
                                end

                                end
                                if @src_cat_name = "BILLING" set @camp_name =
substring(@avail_key,5,2) + " Credit Period " + @campaign_code
                                end
                                if @fulfillment_house = "PALM COAST"
                                begin
                                    if @src_cat_name = "DTP" or @src_cat_name = "GIFT" set
@camp_name = substring(@avail_key, len(@avail_key) - 3, 4) + " Campaign " + @campaign_code
                                    if @src_cat_name = "RENEWALS" set @camp_name =
substring(@avail_key, len(@avail_key) - 3, 4) + " Expire " + @campaign_code
                                    if @src_cat_name = "BILLING" set @camp_name = "Credit
Period " + @campaign_code
                                end
                                if @fulfillment_house = "CDS"
                                begin
                                    if @src_cat_name = "DTP" or @src_cat_name = "GIFT"
                                begin
                                    if substring(@avail_key, len(@avail_key) - 1, 2) < 8
                                begin
                                    set @camp_name = "200" +
substring(@avail_key, len(@avail_key) - 1, 2) + " Campaign " + @campaign_code
                                end
                                    else
                                begin
                                    set @camp_name = "199" +
substring(@avail_key, len(@avail_key) - 1, 2) + " Campaign " + @campaign_code
                                end
                                    --end --if
                                end
                                if @src_cat_name = "RENEWALS" set @camp_name =
@campaign_code
                                if @src_cat_name = "BILLING" set @camp_name =
@campaign_code
                                end

                                if @camp_name is null set @camp_name = "Auto setup Campaign -
Fulfillment House not Set!!!"
                                end --if
                                if @verbose > 0 print "Campaign Name ==> " + @camp_name

                                --set prior source indicator
                                set @i = 0
                                set @prior_source_indicator = ""
                                while @i <= len(@setup_mask)
                                begin
                                    set @i = @i + 1

```

```

        if substring(@setup_mask,@i,1)="Z" set @prior_source_indicator =
@prior_source_indicator + substring(@avail_key, @i,1)
    end --while
    if @verbose < 0 print "Prior Source Indicator ==> " + @prior_source_indicator

--set source name from renewal table
--if the number of schemes is > 1 then the source is split - a match must occur
if @src_cat_name = "RENEWALS" and @number_of_valid_schemes > 1 set
@source_name = null
    if @src_cat_name = "RENEWALS" select @source_name = source_name
        from autosetup_renewal_psdef
        where product_id = @product_id
        and src_cat_name = "RENEWALS"
        and patindex('%' +
substring(@avail_key,1,1) + '%', source_indicator) > 0
        and @dt_valid_to >= valid_from_date
        and @dt_valid_to <= valid_to_date
        --and source_indicator =

substring(@avail_key,1,1)
        --and prior_source_indicator =

@prior_source_indicator
        and patindex('%' + @prior_source_indicator
+ '%', prior_source_indicator) > 0
        if @verbose < 0 print "Source Name ==> " + @source_name
        if @source_name is null and @verbose < 0 print "No match to prior source in
the psdef table!"--if no match was made, move to next scheme
        else
        .begin
        --if a campaign exists use it, if not then create campaign
        set @rec_id = null
        select @rec_id = rec_id
            from campaigns
            where (campaign_code = @campaign_code or camp_name =
@camp_name)
            and src_indicator = @src_indicator
            and src_cat_name = @src_cat_name
            and product_id = @product_id
            and source_name = @source_name
            --and camp_date_fr <= @dt_valid_to
            --and camp_date_to >= @dt_valid_to
        if @rec_id is not null
        select @camp_name = camp_name
            from campaigns
            where rec_id = @rec_id
        if @rec_id is null and @verbose < 0 print "Campaign Not Found!"
        if not (@rec_id is null) and @verbose < 0 print "Campaign Rec ID
==> " + cast(@rec_id as varchar(50))

        select @title_name = title_name
            from corp_info
            where product_id = @product_id
        if @verbose < 0 print "Title Name ==> " + @title_name

        if (@rec_id is null) --no campaign exists, create one; add row to
campaign def

```

```
begin
    insert into campaigns
        select @src_indicator as src_indicator
        , @src_cat_name as src_cat_name
        , @product_id as product_id
        , "Undefined" as camp_type
        , @camp_name as camp_name
        , "Undefined" as camp_status_name
        , null as agt_code
        , null as agt_descr
        , @camp_date_fr as camp_date_fr
        , null as camp_date_to
        , @e_set_name as e_set_name
        , "Undefined" as r_set_name
        , @grp_name as grp_name
        , @title_name as title_name
        , @source_name as source_name
        , null as newstand_sale
        , null as budgeted_gross_pct
        , null as budgeted_vol
        , @campaign_code as campaign_code

    set @rec_id = null
    select @rec_id = rec_id
        from autosetup_campaign_def
        where product_id = @product_id
        and campaign_code = @campaign_code
        and src_cat_name = @src_cat_name
    if @rec_id is null
        insert into autosetup_campaign_def
            (product_id
            , campaign_code
            , campaign_description
            , src_cat_name
            , campaign_date
            , panel_scheme)
            values (@product_id
            , @campaign_code
            , @camp_name
            , @src_cat_name
            , @camp_date_fr
            , null)
    end --if
    --ok, campaign is done, now for the panel
    --Check for panel naming scheme
    set @panel_scheme = null
    set @panel_code = null
    exec concat_p_codes @setup_mask, @avail_key, @panel_code output

    select @panel_scheme = panel_scheme
        from autosetup_campaign_def
        where product_id = @product_id
        and campaign_code = @campaign_code
        --and campaign_description = @campaign_description
        and src_cat_name = @src_cat_name
```

```

--and campaign_date = @camp_date_fr
if @verbose <> 0 print "Panel Scheme ==> " + @panel_scheme

if @panel_scheme is not null and @panel_code is not null --use panel
scheme to determine panel name
begin
    set @panel_name = null
    select @panel_name = panel_name
        from autosetup_panel_def
        where product_id = @product_id
        and panel_scheme = @panel_scheme
        and patindex('%' + @panel_code + '%', panel_codes)
> 0
--gw print statement below
if @verbose <> 0 print "Panel Name ==> " + @panel_name
end
if @panel_scheme is null or (@panel_scheme is not null and
@panel_code is null) or (@panel_scheme is not null and @panel_name is null)--build panel name from
default
begin
    if not(@panel_code > " ") set @panel_name = "Control"
    else set @panel_name = "Panel " + @panel_code
    if @verbose <> 0 print "Panel Name ==> " + @panel_name
end -- if

--check to see if a panel exists
set @rec_id = null
select @rec_id = rec_id
    from panels
    where campaign_code = @campaign_code
    and src_indicator = @src_indicator
    and panel_code = @panel_code
    and src_cat_name = @src_cat_name
    and source_name = @source_name
    and product_id = @product_id
--check for unique panel name
if @rec_id is null select @rec_id = rec_id
    from panels
    where panel_name = @panel_name
    and camp_name = @camp_name
    and src_indicator = @src_indicator
    and source_name = @source_name
    and panel_name = @panel_name
    and src_cat_name = @src_cat_name
    and product_id = @product_id
if @verbose <> 0 Print "Panel ID (null means create a new one) ==> "
+ cast(@rec_id as varchar(50))

if (@rec_id is null) --then panel does not exist so create it
begin
    insert into panels
        select @camp_name as camp_name
        , @src_indicator as src_indicator
        , @panel_name as panel_name
        , @src_cat_name as src_cat_name
        , @product_id as product_id

```

```
, "Undefined" as panel_type
, @e_set_name as e_set_name
, "Undefined" as panel_subtype
, "Undefined" as r_set_name
, "Undefined" as test_type
, null as panel_offer
, null as panel_offer_price
, "Undefined" as panel_pkg
, null as panel_premium
, null as panel_date_start
, null as panel_date_end
, @grp_name as grp_name
, @title_name as title_name
, @source_name as source_name
, "Undefined" as camp_type
, "Undefined" as camp_status_name
, null as agt_code
, null as agt_descr
, @valid_from_date as camp_date_fr
, @valid_to_date as camp_date_to
, null as newstand_sale_marker
, null as newstand_sale
, null as budgeted_gross_pct
, null as budgeted_vol
, @campaign_code as campaign_code
, @panel_code as panel_code
, null as order_qty
, null as dist_qty
```

```
end -if
set @rec_id = null
if @verbose < 0 print "Assigning key to keys table ..."
--OK, now add the key to the keys table
if @verbose < 0 print "Expense set name > " + @e_set_name
if (@e_set_name='0') or (@e_set_name is null) set @e_set_name =
```

"Undefined"

```
insert into keys
select @avail_key as full_promo_key
, @dt_valid_to as dt_valid_to
, @panel_name as panel_name
, @camp_name as camp_name
, @dt_valid_to as dt_valid_from
, @src_indicator as src_indicator
, @src_cat_name as src_cat_name
, @product_id as product_id
, @e_set_name as e_set_name
, null as list_id
, 0 as level1_expense
, null as list_cost_rollout_cpm
, null as list_cost_actual_cpm
, null as fixed_expense
, 'True' as subtotal_flag
, 0 as level2_expense
, "Undefined" as r_set_name
, 0 as level3_expense
```

```

, 0 as level1_revenue
, 0 as level2_revenue
, 0 as level3_revenue
, null as p_key_descr
, "Undefined" as panel_type
, "Undefined" as panel_subtype
, "Undefined" as test_type
, null as panel_offer
, null as panel_offer_price
, "Undefined" as panel_pkg
, null as panel_premium
, null as panel_date_start
, null as panel_date_end
, @grp_name as grp_name
, @title_name as title_name
, @source_name as source_name
, "Undefined" as camp_type
, "Undefined" as camp_status_name
, null as agt_code
, null as agt_descr
, null as camp_date_fr
, null as camp_date_to
, @mail_qty as key_mail_qty
, null mail_qty_override
, null as merge_purge_qty
, "Mail Qty" as list_cost_basis
, 1 as list_cost_factor
, "Undefined" as list_cat_name
, "Undefined" as list_name
, "Undefined" as list_segment_name
, null as optional_effort_id
, null as newstand_sale
, "N" as newstand_sale_marker
, null as budgeted_gross_pct
, null as budgeted_vol
, getdate() as date_assigned
, @campaign_code as campaign_code
, @panel_code as panel_code
, "Auto Setup" as dbusername
, null as order_qty
, null as dist_qty

end -- if
end -- if, no matching source name
if @verbose <> 0 print "Getting next avail key ..."
fetch next from avail_keys into @product_id, @avail_key, @fulfillment_house, @dt_valid_to,
@source_identifier, @mail_qty
end --while
close avail_keys
deallocate avail_keys
delete from avail_keys
where exists
(select *
from keys k
where k.full_promo_key = avail_keys.full_promo_key
and k.product_id = avail_keys.product_id)

```




Procedure Name: Cascade_update

```
@field_name VARCHAR(100),
@old_field_value VARCHAR(200),
@new_field_value VARCHAR(200),
@level VARCHAR(30),
@product_id Numeric(9)
AS
declare @sql NVarchar(200)

--print @field_name + " " + @field_value
IF @level = "KEYS" or @level = "PANELS" or @level = "CAMPAIGNS"
begin
    set @sql = 'UPDATE keys SET ' + @field_name + ' = @new_value where ' + @field_name + ' =
    @old_value '
    if @product_id > -1
    begin
        set @sql = @SQL + ' and Product_id = ' + cast(@product_id as VARCHAR)
    end
    exec sp_executesql @Sql, N'@new_value VARCHAR(200), @old_value
    VARCHAR(200)',@new_field_value,@old_field_value
    END

IF @level = "PANELS" or @level = "CAMPAIGNS"
BEGIN
    set @sql = 'UPDATE panels SET ' + @field_name + ' = @new_value where ' + @field_name + ' =
    @old_value '
    if @product_id > -1
    begin
        set @sql = @SQL + ' and Product_id = ' + cast(@product_id as VARCHAR)
    end
    exec sp_executesql @Sql, N'@new_value VARCHAR(200), @old_value
    VARCHAR(200)',@new_field_value,@old_field_value
    END

IF @level = "CAMPAIGNS"
BEGIN
    set @sql = 'UPDATE campaigns SET ' + @field_name + ' = @new_value where ' + @field_name + ' =
    @old_value '
    if @product_id > -1
    begin
        set @sql = @SQL + ' and Product_id = ' + cast(@product_id as VARCHAR)
    end
    exec sp_executesql @Sql, N'@new_value VARCHAR(200), @old_value
    VARCHAR(200)',@new_field_value,@old_field_value
    END
```


Procedure Name: Concat_c_codes

```
@setup_mask varchar(50)
, @avail_key varchar(50)
, @campaign_code varchar(50) output as
```

```
--local vars
declare @i int
```

```
begin
```

```
    set @i = 0
    set @campaign_code = ""
    while @i < len(@setup_mask) --loop through each char in mask and add to camp code where
mask letter is a C
    begin
        set @i = @i + 1
        --select @i, substring(@setup_mask, @i, 1), substring(@avail_key, @i, 1)
        if substring(@setup_mask, @i, 1) = "C" set @campaign_code = @campaign_code +
substring(@avail_key, @i, 1)
    end --while
    --return @campaign_code
end --concat c codes
```



Procedure Name: concat_p_codes

```
@setup_mask varchar(50)
, @avail_key varchar(50)
, @panel_name varchar(50) output as
```

```
--local vars
declare @i int
```

```
begin
```

```
    set @i = 0
    set @panel_name = ""
    print "Entering concat p codes..."
    print "setup mask is ==> " + @setup_mask
    print "Avail key is ==> " + @avail_key
    --print "panel name is ==> " + @panel_name
    while @i < len(@setup_mask) --loop through each char in mask and add to panel name where
    mask letter is a P
    begin
        set @i = @i + 1
        --select @i, substring(@setup_mask, @i, 1), substring(@avail_key, @i, 1)
        if substring(@setup_mask, @i, 1) = "P" set @panel_name = @panel_name +
        substring(@avail_key, @i, 1)
        --print "Panel is now ==> " + @panel_name
    end --while
    print "Panel is now ==> " + @panel_name
end --concat p codes
```

Procedure Name: Create_r101_topline

```
truncate table r101_topline
INSERT INTO r101_topline
Select  keys.Product_id as Product_id,
        keys.src_cat_name as src_cat_name,
        keys.source_name as source_name,
        keys.camp_name as camp_name,
        -- summary fields
        sum(r101.gross_subs) AS gross_subs,
        sum(r101.tot_net_subs) AS tot_net_subs,
        min(keys.camp_date_fr) AS camp_date_fr,
        max(keys.budgeted_gross_pct) AS Budgeted_gross_pct,
        max(keys.Budgeted_vol) AS Budgeted_vol,
        sum(r101.mail_qty) promo_qty
FROM    keys keys, r101 r101, est_max_effort est_max_effort
WHERE   keys.full_promo_key = r101.r101_key
AND     keys.product_id = r101.product_id
-- keys to r101
AND r101.product_id = est_max_effort.product_id
AND r101.source_IDENTIFIER = est_max_effort.souce_IDENTIFIER
AND r101.issue_IDENTIFIER = est_max_effort.issue_IDENTIFIER
-- other stuff
AND r101.prior_source_IDENTIFIER = est_max_effort.prior_source_IDENTIFIER
and keys.src_cat_name = 'Renewals'
AND r101.effort_identifier = est_max_effort.max_effort
GROUP BY keys.Product_id,
        keys.src_cat_name,
        keys.source_name,
        keys.camp_name

Insert into r101_topline
-- else case ....

Select  keys.Product_id as Product_id,
        keys.src_cat_name as src_cat_name,
        keys.source_name as source_name,
        keys.camp_name as camp_name,
        -- summary fields
        sum(r101.gross_subs) AS gross_subs,
        sum(r101.tot_net_subs) AS tot_net_subs,
        min(keys.camp_date_fr) AS camp_date_fr,
        max(keys.budgeted_gross_pct) AS Budgeted_gross_pct,
        max(keys.Budgeted_vol) AS Budgeted_vol,
        0 promo_qty
FROM    keys keys, r101 r101, est_max_effort est_max_effort
WHERE   keys.full_promo_key = r101.r101_key
AND     keys.product_id = r101.product_id
-- keys to r101
AND r101.product_id = est_max_effort.product_id
AND r101.source_IDENTIFIER = est_max_effort.souce_IDENTIFIER
AND r101.issue_IDENTIFIER = est_max_effort.issue_IDENTIFIER
AND r101.prior_source_IDENTIFIER = est_max_effort.prior_source_IDENTIFIER
-- other stuff
```



```
and keys.src_cat_name = 'Renewals'
AND r101.effort_identifier <> est_max_effort.max_effort
GROUP BY keys.Product_id,
         keys.src_cat_name,
         keys.source_name,
         keys.camp_name
--
INSERT INTO R101_TOPLINE
Select  keys.Product_id as Product_id,
        keys.src_cat_name as src_cat_name,
        keys.source_name as source_name,
        keys.camp_name as camp_name,
        -- summary fields
        sum(r101.gross_subs) AS gross_subs,
        sum(r101.tot_net_subs) AS tot_net_subs,
        min(keys.camp_date_fr) AS camp_date_fr,
        max(keys.budgeted_gross_pct) AS Budgeted_gross_pct,
        max(keys.Budgeted_vol) AS Budgeted_vol,
        sum(newstand_sale) promo_qty
FROM    keys keys, r101 r101, est_max_effort est_max_effort
WHERE   keys.full_promo_key = r101.r101_key
AND     keys.product_id = r101.product_id
-- keys to r101
AND r101.product_id = est_max_effort.product_id
AND r101.source_IDENTIFIER = est_max_effort.souce_IDENTIFIER
AND r101.issue_IDENTIFIER = est_max_effort.issue_IDENTIFIER
AND r101.prior_source_IDENTIFIER = est_max_effort.prior_source_IDENTIFIER
-- other stuff
and keys.src_cat_name <> 'Renewals'
AND keys.subtotal_flag = 'true'
and Newstand_sale_marker = 'Y'
GROUP BY keys.Product_id,
         keys.src_cat_name,
         keys.source_name,
         keys.camp_name
INSERT INTO R101_TOPLINE
Select  keys.Product_id as Product_id,
        keys.src_cat_name as src_cat_name,
        keys.source_name as source_name,
        keys.camp_name as camp_name,
        -- summary fields
        sum(r101.gross_subs) AS gross_subs,
        sum(r101.tot_net_subs) AS tot_net_subs,
        min(keys.camp_date_fr) AS camp_date_fr,
        max(keys.budgeted_gross_pct) AS Budgeted_gross_pct,
        max(keys.Budgeted_vol) AS Budgeted_vol,
        sum(newstand_sale) promo_qty
FROM    keys keys, r101 r101, est_max_effort est_max_effort
WHERE   keys.full_promo_key = r101.r101_key
AND     keys.product_id = r101.product_id
-- keys to r101
AND r101.product_id = est_max_effort.product_id
AND r101.source_IDENTIFIER = est_max_effort.souce_IDENTIFIER
AND r101.issue_IDENTIFIER = est_max_effort.issue_IDENTIFIER
AND r101.prior_source_IDENTIFIER = est_max_effort.prior_source_IDENTIFIER
```

```
-- other stuff
and keys.src_cat_name <> 'Renewals'
AND keys.subtotal_flag = 'true'
and Newstand_sale_marker = 'Y'
GROUP BY keys.Product_id,
        keys.src_cat_name,
        keys.source_name,
        keys.camp_name

INSERT INTO R101_TOPLINE
Select  keys.Product_id as Product_id,
        keys.src_cat_name as src_cat_name,
        keys.source_name as source_name,
        keys.camp_name as camp_name,
        -- summary fields
        sum(r101.gross_subs) AS gross_subs,
        sum(r101.tot_net_subs) AS tot_net_subs,
        min(keys.camp_date_fr) AS camp_date_fr,
        max(keys.budgeted_gross_pct) AS Budgeted_gross_pct,
        max(keys.Budgeted_vol) AS Budgeted_vol,
        0 promo_qty
FROM    keys keys, r101 r101, est_max_effort est_max_effort
WHERE   keys.full_promo_key = r101.r101_key
AND     keys.product_id = r101.product_id
-- keys to r101
AND r101.product_id = est_max_effort.product_id
AND r101.source_IDENTIFIER = est_max_effort.souce_IDENTIFIER
AND r101.issue_IDENTIFIER = est_max_effort.issue_IDENTIFIER
AND r101.prior_source_IDENTIFIER = est_max_effort.prior_source_IDENTIFIER
-- other stuff
and keys.src_cat_name <> 'Renewals'
AND keys.subtotal_flag = 'true'
and Newstand_sale_marker = 'N'
GROUP BY keys.Product_id,
        keys.src_cat_name,
        keys.source_name,
        keys.camp_name
```

```
INSERT INTO R101_TOPLINE
Select  keys.Product_id as Product_id,
        keys.src_cat_name as src_cat_name,
        keys.source_name as source_name,
        keys.camp_name as camp_name,
        -- summary fields
        sum(r101.gross_subs) AS gross_subs,
        sum(r101.tot_net_subs) AS tot_net_subs,
        min(keys.camp_date_fr) AS camp_date_fr,
        max(keys.budgeted_gross_pct) AS Budgeted_gross_pct,
        max(keys.Budgeted_vol) AS Budgeted_vol,
        sum(keys.mail_qty_override) promo_qty
FROM    keys keys, r101 r101, est_max_effort est_max_effort
WHERE   keys.full_promo_key = r101.r101_key
AND     keys.product_id = r101.product_id
```

```
-- keys to r101
AND r101.product_id = est_max_effort.product_id
AND r101.source_IDENTIFIER = est_max_effort.souce_IDENTIFIER
AND r101.issue_IDENTIFIER = est_max_effort.issue_IDENTIFIER
AND r101.prior_source_IDENTIFIER = est_max_effort.prior_source_IDENTIFIER
-- other stuff
and keys.src_cat_name <> 'Renewals'
AND keys.subtotal_flag = 'false'
and keys.mail_qty_override > 0
GROUP BY keys.Product_id,
        keys.src_cat_name,
        keys.source_name,
        keys.camp_name
```

INSERT INTO R101_TOPLINE

```
Select  keys.Product_id as Product_id,
        keys.src_cat_name as src_cat_name,
        keys.source_name as source_name,
        keys.camp_name as camp_name,
        -- summary fields
        sum(r101.gross_subs) AS gross_subs,
        sum(r101.tot_net_subs) AS tot_net_subs,
        min(keys.camp_date_fr) AS camp_date_fr,
        max(keys.budgeted_gross_pct) AS Budgeted_gross_pct,
        max(keys.Budgeted_vol) AS Budgeted_vol,
        sum(keys.key_mail_qty) promo_qty
FROM    keys keys, r101 r101, est_max_effort est_max_effort
WHERE   keys.full_promo_key = r101.r101_key
AND     keys.product_id = r101.product_id
-- keys to r101
AND r101.product_id = est_max_effort.product_id
AND r101.source_IDENTIFIER = est_max_effort.souce_IDENTIFIER
AND r101.issue_IDENTIFIER = est_max_effort.issue_IDENTIFIER
AND r101.prior_source_IDENTIFIER = est_max_effort.prior_source_IDENTIFIER
-- other stuff
and keys.src_cat_name <> 'Renewals'
AND keys.subtotal_flag = 'false'
and keys.mail_qty_override <= 0
GROUP BY keys.Product_id,
        keys.src_cat_name,
        keys.source_name,
        keys.camp_name
```

Procedure Name: del_key

```
/****** Object: Stored Procedure dbo.del_Key  Script Date: 1/14/99 3:43:07 PM *****/  
/****** Object: Stored Procedure dbo.del_Key  Script Date: 1/12/99 11:05:37 PM *****/  
CREATE PROCEDURE del_Key  
    @full_promo_key varchar (25),@product_id numeric(8,2)  
AS  
delete from keys where product_id=@product_id and full_promo_key = @full_promo_key
```

Procedure Name: do_101flow_selDatefix

```

    @product_id numeric(9,2),      -- product_id
    @Promo_Key varchar(25),        -- promo key being inserted
    @reorg_date smalldatetime,     -- the reorg date from the file that the key comes from
    @import_file_type char(1) -- the type of import file i.e. new business/cds , etc.

AS
declare
    --execute do101_flow_selDateFix @product_id, @Promo_Key, @reorg_date
    @nPromo_Key varchar(25),      -- promo_key minus the right 6 characters + @maxSel_Date +
*
    @ePromo_Key varchar(25),      -- existing modified promo key from a prior calculation
    @Key_Length integer,
    @maxSel_Date smalldatetime    -- max select date if prior same key exists
select @nPromo_Key = len(@Promo_Key) --promo_key length
If (@import_file_type = 'C')
    BEGIN
        declare c_flow cursor for
        select max(select_date) m, r101_key from r101_flow where
        product_id = @product_id and
        r101_key like (substring(@Promo_Key,1,@nPromo_Key-9) + '[ ]G[ ]%') and
        (len(r101_key) > 17 OR IMPORT_file_type = 'C') group by select_date, r101_key order by m desc
        print 'searching for key match on C imp file type'
    END
    else
    BEGIN
        declare c_flow cursor for
        select max(select_date) m, r101_key from r101_flow where
        product_id = @product_id and r101_key like (substring(@Promo_Key,1,@nPromo_Key-6) + '%')
        and len(r101_key) < 18 group by select_date, r101_key order by m desc
    END
open c_flow
fetch c_flow into @maxSel_Date, @ePromo_Key
--print 'fetch_status=' + convert(varchar, @@fetch_status)
If (@@fetch_status = 0)
    BEGIN
        If (@maxSel_Date > DATEADD(day, -555, getDate()))
            BEGIN
                print convert(varchar, @maxSel_Date) + 'fetch status = match found(0)'
                update r101_flow set select_date = @maxSel_Date, r101_key = @ePromo_Key
                where product_id = @product_id and r101_key = @Promo_Key
                close c_flow
                deallocate c_flow
                return
            END
    ELSE -- no prior matching keys or select date is null
        BEGIN
            print 'appending date to key'
            select @maxSel_Date = @reorg_date
            update r101_flow set select_date = @maxSel_Date,
            r101_key = (substring(@Promo_Key,1,@nPromo_Key-6) + CONVERT (varchar,
            @maxSel_Date , 101)+'**')
            where product_id = @product_id and r101_key = @Promo_Key
            close c_flow

```




END

```
close c_flow
deallocate c_flow
return
```

Procedure Name: doavail401keys

```

begin --sp
declare @r401_key varchar(25), @select_date smalldatetime, @bills_mailed numeric(12,2), @product_id
numeric(8,0), @full_promo_key varchar(25), @dt_valid_to smalldatetime, @source_identifier
char(1), @keyCount numeric(8,0), @import_file_type char(1)
declare @fulfillment_house varchar(50)

set nocount on
/*add for product id smarts */
declare c_r401 cursor for /* r401 recordset - the main/reference one */
select r401_key, select_date, bills_mailed, product_id, source_identifier, import_file_type from
r401_summary where not exists
(select * from keys k
where k.full_promo_key = r401_summary.r401_key and k.product_id =
r401_summary.product_id) order by product_id, r401_key
open c_r401
fetch c_r401 into @r401_key, @select_date, @bills_mailed, @product_id, @source_identifier,
@import_file_type

while (@@fetch_status = 0)
begin
Insert into avail_keys (full_promo_key --1
,dt_valid_to --2
,mail_qty --3
,product_id --4
,source_identifier --5
,position1 --6
,position2 --7
,position3 --8
,position4 --9
,position5 --10
,position6 --11
,position7 --12
,position8 --13
,position9 --14
,position10 --15
,fulfillment_house ) --16
values
(@r401_key --1
,ISNULL(@select_date, getDate()) --2
,@bills_mailed --3
,@product_id --4
,@source_identifier --5
,SUBSTRING(@r401_key, 1, 1) --6
,SUBSTRING(@r401_key, 2, 1) --7
,SUBSTRING(@r401_key, 3, 1) --8
,SUBSTRING(@r401_key, 4, 1) --9
,SUBSTRING(@r401_key, 5, 1) --10
,SUBSTRING(@r401_key, 6, 1) --11
,SUBSTRING(@r401_key, 7, 1) --12
,SUBSTRING(@r401_key, 8, 1) --13
,SUBSTRING(@r401_key, 9, 1) --14
,SUBSTRING(@r401_key, 10, 1) --15

```

```
, case @import_file_type
  when 'A' then 'CENTROBE'
  when 'B' then 'CENTROBE'
  when 'C' then 'CENTROBE'
  when 'E' then 'CENTROBE'
  when 'F' then 'CDS'
  when 'G' then 'CDS'
  when 'H' then 'CDS'
  when 'I' then 'PALM COAST'
  when 'J' then 'PALM COAST'
  when 'K' then 'PALM COAST'
  else null
end)
```

```
  fetch c_r401 into @r401_key,@select_date,@bills_mailed,@product_id,@source_identifier,
@import_file_type
  end
close c_r401
deallocate c_r401
```

```
delete from avail_keys where exists
  (select * from keys k
   where k.full_promo_key = avail_keys.full_promo_key and k.product_id = avail_keys.product_id)
end --sp
```

```
exec autosetup_wizard 1, -1 -1 to display text (zero suppresses), -1 for all products
```



Procedure Name: doavailkeysbypid

@PID numeric(9,0)

AS

```
declare @r101_key varchar(25), @select_date smalldatetime, @mail_qty numeric(12,2), @product_id
numeric(8,0), @full_promo_key varchar(25), @dt_valid_to smalldatetime, @source_identifier
char(1), @keyCount numeric(8,0)
set nocount on
/*add for product id smarts */
declare c_r101 cursor for /* r101 recordset - the main/reference one */
    select distinct r101_key, select_date, mail_qty, product_id, source_identifier from r101 where
product_id = @PID order by product_id
/*declare c_keys cursor for
    select count(full_promo_key) from keys where product_id=@product_id and full_promo_key =
    @r101_key*/

open c_r101
fetch c_r101 into @r101_key, @select_date, @mail_qty, @product_id, @source_identifier
delete from avail_keys where product_id = @PID /*delete all existing avail_keys from avail_keys list*/

while (@@fetch_status = 0)
    begin
        Insert into avail_keys (full_promo_key, dt_valid_to, mail_qty, product_id, source_identifier,
            position1, position2, position3, position4, position5, position6, position7,
            position8, position9, position10) values

        (@r101_key, DATEADD(month, 20, ISNULL(@select_date, getDate())), @mail_qty, @product_id, @source_
            identifier,
            SUBSTRING(@r101_key, 1, 1), SUBSTRING(@r101_key, 2, 1), SUBSTRING(@r101_key, 3, 1),
            SUBSTRING(@r101_key, 4, 1), SUBSTRING(@r101_key, 5, 1),
            SUBSTRING(@r101_key, 6, 1), SUBSTRING(@r101_key, 7, 1), SUBSTRING(@r101_key, 8, 1),
            isnull(SUBSTRING(@r101_key, 9, 1), null), ISNULL(SUBSTRING(@r101_key, 10, 1), null) )

        fetch c_r101 into @r101_key, @select_date, @mail_qty, @product_id, @source_identifier
        end
close c_r101
deallocate c_r101

delete from avail_keys where exists
    (select * from keys k
    where k.full_promo_key = avail_keys.full_promo_key and k.product_id = avail_keys.product_id)
print 'Attempting to Execute doAvail401KeysByPID'
execute doavail401keysbypid @PID
return
```

Procedure Name: dovail401keysbypid

@PID numeric(9,0)
AS

declare @r401_key varchar(25), @select_date smalldatetime, @bills_mailed numeric(12,2), @product_id
numeric(8,0), @full_promo_key varchar(25), @dt_valid_to smalldatetime, @source_identifier
char(1), @keyCount numeric(8,0)

set nocount on

/*add for product id smarts */

declare c_r401 cursor for /* r401 recordset - the main/reference one */

select distinct r401_key, select_date, bills_mailed, product_id, source_identifier from r401_summary
where product_id = @PID order by product_id

open c_r401

fetch c_r401 into @r401_key, @select_date, @bills_mailed, @product_id, @source_identifier

while (@@fetch_status = 0)
begin

Insert into avail_keys (full_promo_key, dt_valid_to, mail_qty, product_id, source_identifier,
position1, position2, position3, position4, position5, position6, position7,
position8, position9, position10) values

(@r401_key, DATEADD(month, 20, ISNULL(@select_date, getDate())), @bills_mailed, @product_id, @sour
ce_identifier,

SUBSTRING(@r401_key, 1, 1), SUBSTRING(@r401_key, 2, 1), SUBSTRING(@r401_key, 3, 1),
SUBSTRING(@r401_key, 4, 1), SUBSTRING(@r401_key, 5, 1),
SUBSTRING(@r401_key, 6, 1), SUBSTRING(@r401_key, 7, 1), SUBSTRING(@r401_key, 8, 1),
isnull(SUBSTRING(@r401_key, 9, 1), null), ISNULL(SUBSTRING(@r401_key, 10, 1), null))

fetch c_r401 into @r401_key, @select_date, @bills_mailed, @product_id, @source_identifier
end

close c_r401

deallocate c_r401

delete from avail_keys where exists

(select * from keys k

where k.full_promo_key = avail_keys.full_promo_key and k.product_id = avail_keys.product_id)

return

Procedure Name: dovailkeys

```
declare @r101_key varchar(25), @fulfillment_house varchar(50), @select_date smalldatetime, @mail_qty
numeric(12,2), @product_id numeric(8,0), @full_promo_key varchar(25), @dt_valid_to
smalldatetime, @source_identifier char(1), @keyCount numeric(8,0)
set nocount on
/*add for product id smarts */

declare c_r101 cursor for /* r101 recordset - the main/reference one */
select r101_key, select_date, mail_qty, product_id, source_identifier, fulfillment_house from r101 where not
exists
    (select * from keys k
     where k.full_promo_key = r101.r101_key and k.product_id = r101.product_id) order by
product_id, r101_key
/*declare c_keys cursor for
select count(full_promo_key) from keys where product_id=@product_id and full_promo_key =
@r101_key*/

open c_r101
fetch c_r101 into @r101_key, @select_date, @mail_qty, @product_id, @source_identifier,
@fulfillment_house
truncate table avail_keys /*delete all existing avail_keys from avail_keys list*/

while (@@fetch_status = 0)
begin
    --print "inserting into avail_keys ==> " + @r101_key
    Insert into avail_keys (full_promo_key, dt_valid_to, mail_qty, product_id, source_identifier,
    position1, position2, position3, position4, position5, position6, position7,
    position8, position9, position10, fulfillment_house) values
    (@r101_key, ISNULL(@select_date, getDate()), @mail_qty, @product_id, @source_identifier,
    SUBSTRING(@r101_key, 1, 1), SUBSTRING(@r101_key, 2, 1), SUBSTRING(@r101_key, 3, 1),
    SUBSTRING(@r101_key, 4, 1), SUBSTRING(@r101_key, 5, 1),
    SUBSTRING(@r101_key, 6, 1), SUBSTRING(@r101_key, 7, 1), SUBSTRING(@r101_key, 8, 1),
    SUBSTRING(@r101_key, 9, 1), SUBSTRING(@r101_key, 10, 1), @fulfillment_house)

    fetch c_r101 into
    @r101_key, @select_date, @mail_qty, @product_id, @source_identifier, @fulfillment_house
end
close c_r101
deallocate c_r101

delete from avail_keys where exists
    (select * from keys k
     where k.full_promo_key = avail_keys.full_promo_key and k.product_id = avail_keys.product_id)

--exec autosetup_wizard
exec doavail401keys
return
```

Procedure Name: docampesets

```
declare c_camp cursor for
    select product_id,camp_name, e_set_name from campaigns where e_set_name > ''

declare @product_id decimal(9), @camp_name varchar(250), @e_set_name varchar (50)
open c_camp
fetch c_camp into @product_id, @camp_name, @e_set_name

while (@@fetch_status = 0)
    begin
        /*UPDATE panels SET e_set_name = @e_set_name WHERE CURRENT OF c_panel*/
        update keys set e_set_name=@e_set_name where product_id = @product_id and camp_name =
        @camp_name and (keys.e_set_name < '' or keys.e_set_name is null)
        fetch c_camp into @product_id, @camp_name, @e_set_name
    end

close c_camp
deallocate c_camp

return
```

Procedure Name: docamprsets

```
declare c_campr cursor for
    select product_id,camp_name, r_set_name from campaigns where r_set_name > ''

declare @product_id decimal(9), @camp_name varchar(250), @r_set_name varchar (50)
open c_campr
fetch c_campr into @product_id, @camp_name, @r_set_name

while (@@fetch_status = 0)
    begin
        /*UPDATE panels SET e_set_name = @e_set_name WHERE CURRENT OF c_panel*/
        update keys set r_set_name=@r_set_name where product_id = @product_id and camp_name =
        @camp_name and (keys.r_set_name < '' or keys.r_set_name is null)
        fetch c_campr into @product_id, @camp_name, @r_set_name
    end

close c_campr
deallocate c_campr

return
```


Procedure Name: doestmaxeffort

```
begin
truncate table est_max EFFORT
INSERT INTO est_max EFFORT
select est_max1.product_id,
       est_max1.source_identifier,
       est_max1.issue_identifier,
       est_max1.prior_source_identifier,
       min(est_max1.effort_identifier),
       min(est_max1.sum_mail_qty)
FROM v_est_max1 est_max1, v_est_max2 est_max2
WHERE est_max1.sum_mail_qty = est_max2.est_max
and est_max1.source_identifier = est_max2.source_identifier
and est_max1.issue_identifier=est_max2.issue_identifier
and est_max1.prior_source_identifier = est_max2.prior_source_identifier
GROUP by est_max1.product_id,
         est_max1.source_identifier,
         est_max1.issue_identifier,
         est_max1.prior_source_identifier
end

return
```



```
CREATE PROCEDURE doMW AS
declare @product_id numeric(8,0), @stat_name varchar(50)
/* */
declare c_ cursor for
    select product_id from corp_info
declare c_work cursor for
    select r_set_name from revenue_sets where product_id = @product_id and r_set_name =
'Undefined'
open c_
fetch c_ into @product_id

delete from revenue_sets where r_set_name = 'Undefined'
delete from revenue_sets where r_set_name = 'UNDEFINED'

while (@@fetch_status = 0)
begin
    open c_work
    fetch c_work into @stat_name
    If (@@fetch_status <> 0)
    begin
        Insert into revenue_sets (product_id, r_set_name) values (@product_id, 'Undefined')

        close c_work
    end
    else
    begin
        close c_work
    end
end
fetch c_ into @product_id
end
close c_
deallocate c_
deallocate c_work

update campaigns set r_set_name = 'Undefined' where r_set_name is null
update campaigns set r_set_name = 'Undefined' where r_set_name = "
update panels set r_set_name = 'Undefined' where r_set_name is null
update panels set r_set_name = 'Undefined' where r_set_name = "
update keys set r_set_name = 'Undefined' where r_set_name is null
update keys set r_set_name = 'Undefined' where r_set_name = "
return
```

Procedure Name: doPanelESets

```
declare c_panel cursor for
    select product_id, panel_name, e_set_name from panels where e_set_name > ''

declare @product_id decimal(9), @panel_name varchar(250), @e_set_name varchar(50)
open c_panel
fetch c_panel into @product_id, @panel_name, @e_set_name

while (@@fetch_status = 0)
begin
    /*UPDATE panels SET e_set_name = @e_set_name WHERE CURRENT OF c_panel*/
    update keys set e_set_name = @e_set_name where product_id = @product_id and panel_name =
    @panel_name and (keys.e_set_name < '' or keys.e_set_name is null)
    fetch c_panel into @product_id, @panel_name, @e_set_name
end

close c_panel
deallocate c_panel

return
```



Procedure Name: doPanelRsets

```
declare c_panelr cursor for
    select product_id,panel_name, r_set_name from panels where r_set_name > ''

declare @product_id decimal(9), @panel_name varchar(250), @r_set_name varchar (50)
open c_panelr
fetch c_panelr into @product_id, @panel_name, @r_set_name

while (@@fetch_status = 0)
begin
    /*UPDATE panels SET e_set_name = @e_set_name WHERE CURRENT OF c_panel*/
    update keys set r_set_name=@r_set_name where product_id = @product_id and panel_name =
    @panel_name and (keys.r_set_name < '' or keys.r_set_name is null)
    fetch c_panelr into @product_id, @panel_name, @r_set_name
end

close c_panelr
deallocate c_panelr

return
```

Procedure Name: doProdSetup

```

/***** Object: Stored Procedure dbo.doProdSetup  Script Date: 1/14/99 3:43:07 PM *****/
/***** Object: Stored Procedure dbo.doProdSetup  Script Date: 1/12/99 11:05:37 PM *****/
/***** Object: Stored Procedure dbo.doProdSetup  Script Date: 11/28/98 2:31:13 PM *****/
/***** Object: Stored Procedure dbo.doProdSetup  Script Date: 11/17/98 2:43:36 PM *****/

declare @product_id numeric(8,0), @grp_name varchar(50), @title_name varchar(50), @src_prod_id
numeric(8,0)
/**/
declare c_corp_info cursor for
    select product_id,grp_name,title_name from corp_info
declare c_src_category cursor for
    select product_id from src_category where product_id = @product_id
open c_corp_info
fetch c_corp_info into @product_id,@grp_name,@title_name
/*if (@@fetch_status <> 0)
begin
end*/
while (@@fetch_status = 0)
begin
    open c_src_category
    fetch c_src_category into @src_prod_id
    If (@@fetch_status <> 0)
    begin
        Insert into src_category (product_id,src_cat_name,src_cat_descr,grp_name,title_name)
values (@product_id,'Agents',",",@grp_name,@title_name)
        Insert into src_category (product_id,src_cat_name,src_cat_descr,grp_name,title_name)
values (@product_id,'Renewals','R things',@grp_name,@title_name)
        Insert into src_category (product_id,src_cat_name,src_cat_descr,grp_name,title_name)
values (@product_id,'Billing',",",@grp_name,@title_name)
        Insert into src_category (product_id,src_cat_name,src_cat_descr,grp_name,title_name)
values (@product_id,'DTP',",",@grp_name,@title_name)
        Insert into src_category (product_id,src_cat_name,src_cat_descr,grp_name,title_name)
values (@product_id,'Gifts',",",@grp_name,@title_name)
        Insert into src_category (product_id,src_cat_name,src_cat_descr,grp_name,title_name)
values (@product_id,'Renewals',",",@grp_name,@title_name)
        Insert into src_category (product_id,src_cat_name,src_cat_descr,grp_name,title_name)
values (@product_id,'UNNASSIGNED',",",@grp_name,@title_name)
        Insert into source
(product_id,src_cat_name,source_name,src_indicator,grp_name,title_name) values
(@product_id,'Renewals','RenewalsSRC','R',@grp_name,@title_name)
        close c_src_category
    end
    else
    begin
        close c_src_category
    end
    fetch c_corp_info into @product_id,@grp_name,@title_name
end
close c_corp_info
deallocate c_corp_info
deallocate c_src_category
return
```





Procedure Name: doR101_summary

```
begin
truncate table r101

--update r101_flow set select_date = getDate() where select_date < '1901-01-01'

/*
update r101_flow set select_date = getDate() where not exists
(select * from r101_flow r
where r.r101_key = r101_flow.r101_key and r.product_id = r101_flow.product_id and select_date
> '1990-01-01')
*/

Insert into r101
(r101_reorg_date,
product_id,
r101_key,
source_identifier,
issue_identifier,
prior_source_identifier,
effort_identifier,
mail_qty,
select_date,
key_descr,
subs_this_week,
subs_to_date,
gross_subs,
gross_subs_pct,
net_subs,
net_subs_pct,
tot_net_subs,
tot_net_subs_pct,
pct_credit,
pct_cred_renew,
cred_pay_amt,
cred_pay_pct,
avg_net_trm,
avg_net_trm_value,
avg_tot_net_trm,
avg_tot_net_trm_value,
subs_phone,
subs_c_card,
install_1st,
install_1st_total,
subs_gift,
subs_gift_pct_1st_trm,
sub_years_gross,
sub_years_gross_tot_net,
gross_sub_revenue,
net_sub_revenue,
gross_sub_copies,
net_sub_copies,
cash_subs,
```

est_max_eff_mail_qty,
gross_cred_subs,
net_cred_subs,
fulfillment_title_name,
gross_list_rental_qty,
net_list_rental_qty,
package_identifier,
key_descr2,
donees_todate,
total_orders,
tot_order_pct_rspnce,
renew_subs_pct
, fulfillment_house)

EXECUTE("select r101_reorg_date,
product_id,
r101_key,
source_identifier,
issue_identifier,
prior_source_identifier,
effort_identifier,
mail_qty,
select_date,
key_descr,
subs_this_week,
subs_to_date,
gross_subs,
gross_subs_pct,
net_subs,
net_subs_pct,
tot_net_subs,
tot_net_subs_pct,
pct_credit,
pct_cred_renew,
cred_pay_amt,
cred_pay_pct,
avg_net_trm,
avg_net_trm_value,
avg_tot_net_trm,
avg_tot_net_trm_value,
subs_phone,
subs_c_card,
install_1st,
install_1st_total,
subs_gift,
subs_gift_pct_1st_trm,
sub_years_gross,
sub_years_gross_tot_net,
gross_sub_revenue,
net_sub_revenue,
gross_sub_copies,
net_sub_copies,
cash_subs,
est_max_eff_mail_qty,
gross_cred_subs,



```
net_cred_subs,
fulfillment_title_name,
gross_list_rental_qty,
net_list_rental_qty,
package_identifier,
key_descr2,
donees_todate,
total_orders,
tot_order_pct_rspnce,
renew_subs_pct
, isnull(case import_file_type
        when 'A' then 'CENTROBE'
        when 'B' then 'CENTROBE'
        when 'C' then 'CENTROBE'
        when 'E' then 'CENTROBE'
        when 'F' then 'CDS'
        when 'G' then 'CDS'
        when 'H' then 'CDS'
        when 'I' then 'PALM COAST'
        when 'J' then 'PALM COAST'
        when 'K' then 'PALM COAST'
        else null
        end
        ,fulfillment_house)
from vr101_flow_max_dates, r101_flow
where mproduct_id = product_id and
mr101_key=r101_key and mdate=r101_reorg_date
order by mproduct_id, mr101_key, r101_reorg_date")
end
```



Procedure Name: doR401_summary

truncate table r401_summary

update r401_flow set select_date = getDate() where not exists
(select * from r401_flow r
where r.r401_key = r401_flow.r401_key and r.product_id = r401_flow.product_id and select_date
> '1990-01-01')

Insert into r401_summary
(r401_reorg_date,
product_id,
r401_key,
source_identifier,
credit_period_identifier,
current_source_identifier,
effort_identifier,
select_date,
bills_mailed,
bills_mailed_amt,
this_week_pmt,
total_pmt,
pmt_pct,
step_up,
step_up_pct,
pmt_plus_step_up_rev,
avg_rev_per_pmt,
avg_term_per_pmt,
request_cancel,
request_cancel_pct,
per_cancel,
per_cancel_pct,
pmt_on_pmt,
pmt_after_cancel,
total_copies,
fulfillment_title_name,
est_max_eff_mail_qty
, import_file_type)
EXECUTE(
SELECT distinct r401_reorg_date,
product_id,
r401_key,
source_identifier,
credit_period_identifier,
current_source_identifier,
effort_identifier,
select_date,
bills_mailed,
bills_mailed_amt,
this_week_pmt,
total_pmt,
pmt_pct,
step_up,

```
step_up_pct,  
pmt_plus_step_up_rev,  
avg_rev_per_pmt,  
avg_term_per_pmt,  
request_cancel,  
request_cancel_pct,  
per_cancel,  
per_cancel_pct,  
pmt_on_pmt,  
pmt_after_cancel,  
total_copies,  
fulfillment_title_name,  
est_max_eff_mail_qty  
, import_file_type  
from vr401_flow_max_dates, r401_flow where mproduct_id = product_id and  
mr401_key=r401_key and mdate=r401_reorg_date order by product_id, r401_key, r401_reorg_date  
)
```

Procedure Name: doRecycleKey

```
@product_id_r numeric(8,2),
@promo_key_r varchar (25)
AS
```

```
declare @r401_key varchar(25),
        @select_date smalldatetime,
        @mail_qty numeric(12,2),
        @product_id numeric(8,0),
        @full_promo_key varchar(25),
        @dt_valid_to smalldatetime,
        @source_identifier char(1),
        @keyCount numeric(8,0),
        @src_cat_name VARCHAR(50),
        @r401_key varchar(25),
        @bills_mailed numeric(9),
        @fulfillment_house varchar(50),
        @import_file_type varchar(50)
```

```
select @src_cat_name = src_cat_name
from keys where full_promo_key = @promo_key_r
and product_id = @product_id_r
```

```
if @src_cat_name = 'Billing'
```

```
BEGIN
```

```
print @src_cat_name
```

```
declare c_r401 cursor for /* r401 recordset - the main/reference one */
```

```
select r401_key,select_date,bills_mailed,product_id,source_identifier, import_file_type from
r401_summary
```

```
where r401_key = @promo_key_r
```

```
and product_id = @product_id_r
```

```
open c_r401
```

```
fetch c_r401 into @r401_key,@select_date,@bills_mailed,@product_id,@source_identifier,
```

```
@import_file_type
```

```
while (@@fetch_status = 0)
```

```
begin
```

```
Insert into avail_keys (full_promo_key,dt_valid_to,mail_qty,product_id,source_identifier,
```

```
position1,position2,position3,position4,position5,position6, position7,
```

```
position8,position9,position10, fulfillment_house ) values
```

```
(@r401_key,ISNULL(@select_date,getDate()),@bills_mailed,@product_id,@source_identifier,
```

```
SUBSTRING(@r401_key,1,1),SUBSTRING(@r401_key,2,1),SUBSTRING(@r401_key,3,1),
```

```
SUBSTRING(@r401_key,4,1),SUBSTRING(@r401_key,5,1),
```

```
SUBSTRING(@r401_key,6,1),SUBSTRING(@r401_key,7,1),SUBSTRING(@r401_key,8,1),
```

```
isnull(SUBSTRING(@r401_key,9,1), null),
```

```
ISNULL(SUBSTRING(@r401_key,10,1),null)
```

```
, case @import_file_type
```

```
when 'A' then 'CENTROBE'
```

```
when 'B' then 'CENTROBE'
```

```
when 'C' then 'CENTROBE'
```

```
when 'E' then 'CENTROBE'
```

```
when 'F' then 'CDS'
```

```
when 'G' then 'CDS'
```



```
        when 'H' then 'CDS'
        when 'I' then 'PALM COAST'
        when 'J' then 'PALM COAST'
        when 'K' then 'PALM COAST'
    end)
    fetch c_r401 into @r401_key,@select_date,@bills_mailed,@product_id,@source_identifier,
    @import_file_type
    end
close c_r401
deallocate c_r401
--delete from avail_keys where exists
--(select * from keys k
--where k.full_promo_key = avail_keys.full_promo_key and k.product_id =
avail_keys.product_id)
return
END

ELSE

BEGIN
print '101stuff'
set nocount on
/*add for product id smarts */
declare c_r101_r cursor for /* r101 recordset - the main/reference one */
select r101_key,select_date,mail_qty,product_id,source_identifier, fulfillment_house from r101 where
product_id = @product_id_r and r101_key = @promo_key_r
open c_r101_r
fetch c_r101_r into @r101_key,@select_date,@mail_qty,@product_id,@source_identifier,
@import_file_type
while (@@fetch_status = 0)
    begin
        Insert into avail_keys
        (full_promo_key,dt_valid_to,mail_qty,product_id,source_identifier,position1,position2,position3,position4,
        position5,position6, position7, fulfillment_house)
        values
        (@r101_key,ISNULL(@select_date,getDate()),@mail_qty,@product_id,@source_identifier,SUBSTRING
        (@r101_key,1,1),SUBSTRING(@r101_key,2,1),SUBSTRING(@r101_key,3,1),SUBSTRING(@r101_key
        ,4,1),SUBSTRING(@r101_key,5,1),SUBSTRING(@r101_key,6
        ,1),SUBSTRING(@r101_key,7,1)
        ,@import_file_type)
        fetch c_r101_r into @r101_key,@select_date,@mail_qty,@product_id,@source_identifier,
        @import_file_type
    end
close c_r101_r
deallocate c_r101_r
return
END
```

Procedure Name: doRecyclePanelKey

```
@rec_id numeric(8,2)
AS
/* get primary key info */
declare @camp_name      varchar(250),
        @source_name    varchar(50),
        @panel_name     varchar(250),
        @src_cat_name   varchar(30),
        @product_id     numeric(5),
        @keys_product_id numeric(5),
        @full_promo_key  numeric(5)

SELECT      @camp_name = camp_name,
            @source_name = source_name,
            @panel_name = panel_name,
            @src_cat_name = src_cat_name,
            @product_id = product_id
FROM panels
WHERE rec_id = @rec_id

declare c_keys cursor for
select full_promo_key, product_id
from keys
where product_id = @product_id
and camp_name = @camp_name
and source_name = @source_name
and panel_name = @panel_name
and src_cat_name = @src_cat_name

open c_keys
fetch c_keys into @full_promo_key, @keys_product_id

while (@@fetch_status = 0)
begin
    exec doRecycleKey @keys_product_id, @full_promo_key
    fetch c_keys into @full_promo_key, @keys_product_id
end
close c_keys
deallocate c_keys
delete from keys
where product_id = @product_id
and camp_name = @camp_name
and source_name = @source_name
and panel_name = @panel_name
and src_cat_name = @src_cat_name

return
```

Procedure Name: doSourceInds

```
/** creates a src_indicator placeholder from 101 flow ***/
declare @var_src_ind char(1), @var_target_ind char(1)
declare c_r101_ind cursor for
    select distinct source_identifier from r101
declare c_source cursor for
    select src_indicator from source where src_indicator = @var_src_ind
open c_r101_ind
fetch c_r101_ind into @var_src_ind
if (@@fetch_status <> 0)
begin
    print "No Source Indicators found in r101"
    close c_r101_ind
    deallocate c_r101_ind
    return
end
while (@@fetch_status = 0)
begin
    open c_source
    fetch c_source into @var_target_ind
    If (@@fetch_status <> 0)
    begin
        Insert into source (src_cat_name,src_indicator,source_name) values
        ('UNASSIGNED',@var_src_ind,'MISSING')
        close c_source
        end
    else
        begin
            close c_source
            end
    end
    fetch c_r101_ind into @var_src_ind
    end
    close c_r101_ind
    deallocate c_r101_ind
    return
```

Procedure Name: doSummary

```
BEGIN -- only for Palm Coast titles at Primedia
update r401_flow set select_date = r401_reorg_date where select_date = '1900-01-01'
END
```

```
BEGIN
PRINT 'Executing dor101_summary'
end
```

```
BEGIN
execute dor101_summary;
PRINT 'Execution of dor101_summary complete'
END
```

```
BEGIN
PRINT 'Executing dor401_summary'
END
```

```
BEGIN
execute dor401_summary;
PRINT 'Execution of dor401_summary complete'
END
```

```
BEGIN
PRINT 'Executing doavailkeys'
END
```

```
BEGIN
execute doavailkeys;
PRINT 'Execution of doavailkeys complete'
END
```

```
BEGIN
PRINT 'Executing doavail401keys'
END
```

--***** doavail401keys runs when doavailkeys runs !

```
--BEGIN
--execute doavail401keys;
--PRINT 'Execution of doavail401keys complete'
--END
```

```
BEGIN
execute doEstMaxEffort ;
PRINT 'Execution of doavail401keys complete'
END
```

```
Begin
Insert into summary_log values(GetDate(), 'Summary Procedure Run')
Print 'Updated summary log'
```




$\frac{d}{dt} \left(\frac{\partial L}{\partial \dot{x}} \right) = \frac{\partial L}{\partial x}$

Procedure Name: getCorpInfo

```
/****** Object: Stored Procedure dbo.getCorpInfo  Script Date: 1/14/99 3:43:06 PM *****/  
/****** Object: Stored Procedure dbo.getCorpInfo  Script Date: 1/12/99 11:05:37 PM *****/  
(  
  @product_id numeric(8,0)  
)  
AS  
select * from corp_info where product_id = @product_id  
return
```

Procedure Name: expenses

```
/****** Object: Stored Procedure dbo.expenses  Script Date: 1/14/99 3:43:07 PM *****/  
/****** Object: Stored Procedure dbo.expenses  Script Date: 1/12/99 11:05:38 PM *****/  
/****** Object: Stored Procedure dbo.expenses  Script Date: 11/28/98 2:31:13 PM *****/  
/****** Object: Stored Procedure dbo.expenses  Script Date: 11/17/98 2:43:36 PM *****/  
/****** Object: Stored Procedure dbo.expenses  Script Date: 10/19/98 6:03:58 PM *****/
```

```
declare @e_set_name varchar (50),@product_id numeric(8,2),  
        @print_cost numeric(11, 3),@print_cpm_cpu varchar (10),  
        @print_basis varchar (20),  
        @ltr_cost numeric(11, 3),@ltr_cpm_cpu varchar (10),  
        @ltr_basis varchar (20),  
        @postage_out_cost numeric(11, 3),@postage_out_cpm_cpu varchar (10),  
        @postage_out_basis varchar (20),  
        @postage_in_cost numeric(11, 3),@postage_in_cpm_cpu varchar (10),  
        @postage_in_basis varchar (20),  
        @premium_cost numeric(11, 3),@premium_cpm_cpu varchar (10),  
        @premium_basis varchar (20),  
        @badpay_cost numeric(11, 3),@badpay_cpm_cpu varchar (10),  
        @badpay_basis varchar (20),  
        @billing_cost numeric(11, 3),@billing_cpm_cpu varchar (10),  
        @billing_basis varchar (20),  
        @subs_svc_costs numeric(11, 3),@subs_svc_cpm_cpu varchar (10),  
        @subs_svc_basis varchar (20),  
        @exp_fixed_costs numeric(11, 3),@exp_fixed_cpm_cpu varchar (10),  
        @exp_fixed_basis varchar (20),  
        @exp_other_costs numeric(11, 3),@exp_other_cpm_cpu varchar (10),  
        @exp_other_basis varchar (20),  
  
        @full_promo_key varchar (10),  
        @level1_expense numeric(12,3),  
        @level2_expense numeric(12,3),  
        @level3_expense numeric(12,3),  
        @r101_key varchar(10),  
        @mail_qty numeric(12,2),  
        @gross_subs numeric(7,2),  
        @net_subs integer,  
  
        @l1_tmp numeric (12,3),  
        @l1_value numeric (12,3),  
        @l2_tmp numeric (12,3),  
        @l2_value numeric (12,3),  
        @l3_tmp numeric (12,3),  
        @l3_value numeric (12,3)  
  
declare c_exp cursor for  
        select expense_sets.e_set_name, expense_sets.product_id,  
        print_cost,print_cpm_cpu,print_basis,  
        ltr_cost,ltr_cpm_cpu,ltr_basis,  
        /*deleted list_cost,cpm and basis because of redesign-list library activities.. moved to key, still  
need to add code for calculating */  
        postage_out_cost,postage_out_cpm_cpu,postage_out_basis,  
        postage_in_cost,postage_in_cpm_cpu,postage_in_basis,  
        premium_cost,premium_cpm_cpu,premium_basis,
```

```
badpay_cost,badpay_cpm_cpu,badpay_basis,
billing_cost,billing_cpm_cpu,billing_basis,
subs_svc_costs,subs_svc_cpm_cpu,subs_svc_basis,
exp_fixed_costs,exp_fixed_cpm_cpu,exp_fixed_basis,
exp_other_costs,exp_other_cpm_cpu,exp_other_basis,
full_promo_key,
level1_revenue,
level2_revenue,
level3_revenue,
r101_key,
mail_qty,
gross_subs,
net_subs
from expense_sets,keys,r101 where keys.product_id=expense_sets.product_id and
keys.product_id=r101.product_id and
keys.e_set_name = expense_sets.e_set_name and keys.full_promo_key = r101.r101_key
Order by expense_sets.e_set_name
open c_exp
fetch c_exp into
    @e_set_name,@product_id,
    @print_cost,@print_cpm_cpu,
    @print_basis,
    @ltr_cost,@ltr_cpm_cpu,
    @ltr_basis,
    @postage_out_cost,@postage_out_cpm_cpu,
    @postage_out_basis,
    @postage_in_cost,@postage_in_cpm_cpu,
    @postage_in_basis,
    @premium_cost,@premium_cpm_cpu,
    @premium_basis,
    @badpay_cost,@badpay_cpm_cpu,
    @badpay_basis,
    @billing_cost,@billing_cpm_cpu,
    @billing_basis,
    @subs_svc_costs,@subs_svc_cpm_cpu,
    @subs_svc_basis,
    @exp_fixed_costs,@exp_fixed_cpm_cpu,
    @exp_fixed_basis,
    @exp_other_costs,@exp_other_cpm_cpu,
    @exp_other_basis,

    @full_promo_key,
    @level1_expense,
    @level2_expense,
    @level3_expense,
    @r101_key,
    @mail_qty,

    @gross_subs,
    @net_subs
while (@@fetch_status = 0)
begin
select @l1_tmp = 0
select @l1_value = 0
select @l2_tmp = 0
```

```
select @l2_value = 0
select @l3_tmp = 0
select @l3_value = 0
/*****LEVEL 1*****/
If (@print_basis = 'Mail Qty')
begin
select @l1_tmp = (@mail_qty * @print_cost)
end
else if (@print_basis = 'Gross Subs')
begin
select @l1_tmp = (@gross_subs * @print_cost)
end
else if (@print_basis = 'Net Subs')
begin
select @l1_tmp = (@net_subs * @print_cost)
end
if (@print_cpm_cpu = 'CPM')
begin
select @l1_tmp = @l1_tmp / 1000
end
select @l1_value = @l1_value + @l1_tmp

select @l1_tmp = 0
If (@ltr_basis = 'Mail Qty')
begin
select @l1_tmp = (@mail_qty * @ltr_cost)
end
else if (@ltr_basis = 'Gross Subs')
begin
select @l1_tmp = (@gross_subs * @ltr_cost)
end
else if (@ltr_basis = 'Net Subs')
begin
select @l1_tmp = (@net_subs * @ltr_cost)
end
if (@ltr_cpm_cpu = 'CPM')
begin
select @l1_tmp = @l1_tmp / 1000
end
select @l1_value = @l1_value + @l1_tmp

select @l1_tmp = 0
If (@postage_out_basis = 'Mail Qty')
begin
select @l1_tmp = (@mail_qty * @postage_out_cost)
end
else if (@postage_out_basis = 'Gross Subs')
begin
select @l1_tmp = (@gross_subs * @postage_out_cost)
end
else if (@postage_out_basis = 'Net Subs')
begin
select @l1_tmp = (@net_subs * @postage_out_cost)
end
if (@postage_out_cpm_cpu = 'CPM')
```

```
begin

    select @l1_tmp = @l1_tmp / 1000
end
select @l1_value= @l1_value + @l1_tmp
/*****LEVEL 1*****/
/*****LEVEL 2*****/

If (@postage_in_basis = 'Mail Qty')
begin
    select @l2_tmp = (@mail_qty * @postage_in_cost)
end
else if (@postage_in_basis = 'Gross Subs')
begin
    select @l2_tmp = (@gross_subs * @postage_in_cost)
end
else if (@postage_in_basis = 'Net Subs')
begin
    select @l2_tmp = (@net_subs * @postage_in_cost)
end
if (@postage_in_cpm_cpu = 'CPM')
begin
    select @l2_tmp = @l2_tmp / 1000
end
select @l2_value= @l2_value + @l2_tmp

select @l2_tmp = 0
If (@premium_basis = 'Mail Qty')
begin
    select @l2_tmp = (@mail_qty * @premium_cost)
end
else if (@premium_basis = 'Gross Subs')
begin
    select @l2_tmp = (@gross_subs * @premium_cost)
end
else if (@premium_basis = 'Net Subs')
begin
    select @l2_tmp = (@net_subs * @premium_cost)
end
if (@premium_cpm_cpu = 'CPM')
begin
    select @l2_tmp = @l2_tmp / 1000
end
select @l2_value= @l2_value + @l2_tmp
select @l2_tmp = 0
If (@badpay_basis = 'Gross - Net Subs')
begin
    select @l2_tmp = (((@gross_subs - @net_subs) * @badpay_cost)
end
if (@badpay_cpm_cpu = 'CPM')
begin
    select @l2_tmp = @l2_tmp / 1000
end
select @l2_value= @l2_value + @l2_tmp
```

```
select @l2_tmp = 0
If (@billing_basis = 'Mail Qty')
    begin
        select @l2_tmp = (@mail_qty * @billing_cost)
    end
else if (@billing_basis = 'Gross Subs')
    begin
        select @l2_tmp = (@gross_subs * @billing_cost)
    end
else if (@billing_basis = 'Net Subs')
    begin
        select @l2_tmp = (@net_subs * @billing_cost)
    end
if (@billing_cpm_cpu = 'CPM')
    begin
        select @l2_tmp = @l2_tmp / 1000
    end
select @l2_value = @l2_value + @l2_tmp
/*****LEVEL 2*****/
/*****LEVEL 3*****/
If (@subs_svc_basis = 'Mail Qty')
    begin
        select @l3_tmp = (@mail_qty * @subs_svc_costs)
    end
else if (@subs_svc_basis = 'Gross Subs')
    begin
        select @l3_tmp = (@gross_subs * @subs_svc_costs)
    end
else if (@subs_svc_basis = 'Net Subs')
    begin
        select @l3_tmp = (@net_subs * @subs_svc_costs)
    end
if (@subs_svc_cpm_cpu = 'CPM')
    begin
        select @l3_tmp = @l3_tmp / 1000
    end
select @l3_value = @l3_value + @l3_tmp

select @l3_tmp = 0
If (@exp_fixed_basis = 'Mail Qty')
    begin
        select @l3_tmp = (@mail_qty * @exp_fixed_costs)
    end
else if (@exp_fixed_basis = 'Gross Subs')
    begin
        select @l3_tmp = (@gross_subs * @exp_fixed_costs)
    end
else if (@exp_fixed_basis = 'Net Subs')
    begin
        select @l3_tmp = (@net_subs * @exp_fixed_costs)
    end
if (@exp_fixed_cpm_cpu = 'CPM')
    begin
        select @l3_tmp = @l3_tmp / 1000
```

```

        end
select @l3_value= @l3_value + @l3_tmp
        select @l3_tmp = 0
        If (@exp_other_basis = 'Mail Qty')
            begin
                select @l3_tmp = (@mail_qty * @exp_other_costs)
            end
        else if (@exp_other_basis = 'Gross Subs')
            begin
                select @l3_tmp = (@gross_subs * @exp_other_costs)
            end
        else if (@exp_other_basis = 'Net Subs')
            begin
                select @l3_tmp = (@net_subs * @exp_other_costs)
            end
        if (@exp_other_cpm_cpu = 'CPM')
            begin
                select @l3_tmp = @l3_tmp / 1000
            end
        end
select @l3_value= @l3_value + @l3_tmp
/*****LEVEL 3*****/
        UPDATE keys SET level1_expense = @l1_value,level2_expense = @l2_value,level3_expense =
        @l3_value WHERE CURRENT OF c_exp
fetch c_exp into
        @e_set_name,@product_id,
        @print_cost,@print_cpm_cpu,
        @print_basis,
        @ltr_cost,@ltr_cpm_cpu,
        @ltr_basis,
        @postage_out_cost,@postage_out_cpm_cpu,
        @postage_out_basis,
        @postage_in_cost,@postage_in_cpm_cpu,
        @postage_in_basis,
        @premium_cost,@premium_cpm_cpu,
        @premium_basis,
        @badpay_cost,@badpay_cpm_cpu,
        @badpay_basis,
        @billing_cost,@billing_cpm_cpu,
        @billing_basis,
        @subs_svc_costs,@subs_svc_cpm_cpu,
        @subs_svc_basis,
        @exp_fixed_costs,@exp_fixed_cpm_cpu,
        @exp_fixed_basis,
        @exp_other_costs,@exp_other_cpm_cpu,
        @exp_other_basis,

        @full_promo_key,
        @level1_expense,
        @level2_expense,
        @level3_expense,
        @r101_key,
        @mail_qty,
        @gross_subs,
        @net_subs
end

```



```
close c_exp  
deallocate c_exp  
return
```

Procedure Name: maxBillEffort

```
update r401_flow set est_max_eff_mail_qty = bills_mailed where effort_identifier = '1'  
update r401_summary set est_max_eff_mail_qty =bills_mailed where effort_identifier = '1'  
update r401_flow set est_max_eff_mail_qty = 0 where est_max_eff_mail_qty is null  
update r401_summary set est_max_eff_mail_qty = 0 where est_max_eff_mail_qty is null
```

Procedure Name: maxEffort

```
update r101_flow set est_max_eff_mail_qty = mail_qty where effort_identifier = 1
update r101 set est_max_eff_mail_qty = mail_qty where effort_identifier = 1
update r101_flow set est_max_eff_mail_qty = 0 where est_max_eff_mail_qty is null
update r101 set est_max_eff_mail_qty = 0 where est_max_eff_mail_qty is null
```

**Procedure Name:** maxGiftEffort

```
update r301_flow set est_max_eff_mail_qty = mail_qty where effort_identifier = 'I'
update r301_summary set est_max_eff_mail_qty = mail_qty where effort_identifier = 'I'
update r301_flow set est_max_eff_mail_qty = 0 where est_max_eff_mail_qty is null
update r301_summary set est_max_eff_mail_qty = 0 where est_max_eff_mail_qty is null
```

Procedure Name: procedure Rebuild_Rpt_Cats

begin

-- UPDATES IMAGINE REPORTS

truncate table imagine.dbo.rpt_cats

insert into imagine.dbo.rpt_cats

select src_cat_name, src_sub_cat, location, rpt_name, rpt_description, rpt_filters from

pmedia.dbo.rpt_cats

BEGIN

PRINT 'IMAGINE Report Catalog Updated'

END

-- UPDATES IMT PRESENTATION REPORTS

truncate table IMTPresentation.dbo.rpt_cats

insert into IMTPresentation.dbo.rpt_cats

select src_cat_name, src_sub_cat, location, rpt_name, rpt_description, rpt_filters from

pmedia.dbo.rpt_cats

BEGIN

PRINT 'IMT Presentation Report Catalog Updated'

END

-- UPDATES DWELL REPORTS

truncate table dwell.dbo.rpt_cats

insert into dwell.dbo.rpt_cats

select src_cat_name, src_sub_cat, location, rpt_name, rpt_description, rpt_filters from

pmedia.dbo.rpt_cats

BEGIN

PRINT 'Dwell Report Catalog Updated'

END

-- UPDATES DWELL REPORTS

truncate table demo.dbo.rpt_cats

insert into demo.dbo.rpt_cats

select src_cat_name, src_sub_cat, location, rpt_name, rpt_description, rpt_filters from

pmedia.dbo.rpt_cats

BEGIN

PRINT 'Dwell Report Catalog Updated'

END

End



Procedure Name: sp_add_autosetup_scheme

```
@product_id int
, @src_cat_name varchar(50)
, @src_indicator varchar(3)
, @source_name varchar (250)
, @valid_from datetime
, @valid_to datetime
, @setup_mask varchar(50)
, @e_set_name varchar(50)

AS
declare @title varchar(50)
declare @grp_name varchar(50)

select @title = title_name
      , @grp_name = grp_name
      from corp_info
      where product_id = @product_id
--gw added next line
if @e_set_name is null set @e_set_name = 'Undefined'

insert into autosetup_scheme
(
    product_id
  , title
  , grp_name
  , src_cat_name
  , src_indicator
  , source_name
  , valid_from_date
  , valid_to_date
  , setup_mask
  , e_set_name
)
values
(
    @product_id
  , @title
  , @grp_name
  , @src_cat_name
  , @src_indicator
  , @source_name
  , @valid_from
  , @valid_to
  , @setup_mask
  , @e_set_name
)
```

Procedure Name: sp_Add_Campaign

```
@camp_name varchar(250), -- 1
@camp_type varchar(50), -- 2
@camp_status_name varchar(20), -- 3
@camp_date_fr smalldatetime, -- 4
@camp_date_to smalldatetime, -- 5
@agt_code varchar(10), -- 6
@agt_descr varchar(50), -- 7
@e_set_name varchar(50), -- 8
@r_set_name varchar(50), -- 9
@newstand_sale numeric(18), -- 10
@budgeted_gross_pct numeric(9), -- 11
@budgeted_vol numeric(9), -- 12
```

```
@source_name varchar (250), -- 13
@src_indicator char (3), -- 14
@product_id numeric(8,0), -- 15
@src_cat_name varchar(30), -- 16
@grp_name varchar(50), -- 17
@title_name varchar (250) -- 18
```

AS

insert into campaigns

(

```
camp_name,
camp_type,
camp_status_name,
camp_date_fr,
camp_date_to,
agt_code,
agt_descr,
e_set_name,
r_set_name,
newstand_sale,
budgeted_vol,
budgeted_gross_pct,
```

```
product_id,
src_cat_name,
source_name,
src_indicator,
grp_name,
title_name
```

)

values

(

```
@camp_name,
isNull( @camp_type,'Undefined'),
isNull( @camp_status_name,'Undefined'),
@camp_date_fr,
```



@camp_date_to,
@agt_code,
@agt_descr,
isnull(@e_set_name,'Undefined'),
isnull(@r_set_name,'Undefined'),
@newstand_sale,
@budgeted_vol,
@budgeted_gross_pct,

@product_id,
@src_cat_name,
@source_name,
@src_indicator,
@grp_name,
@title_name

)

Procedure Name: sp_add_campaign_def

```
@product_id int
, @campaign_code varchar(50)
, @campaign_description varchar(50)
, @src_cat_name varchar(50)
, @campaign_date datetime
, @panel_scheme varchar(50)
```

AS

insert into autosetup_campaign_def

```
(
    product_id
    , campaign_code
    , campaign_description
    , src_cat_name
    , campaign_date
    , panel_scheme
)
```

values

```
(
    @product_id
    , @campaign_code
    , @campaign_description
    , @src_cat_name
    , @campaign_date
    , @panel_scheme
)
```

Procedure Name: sp_Add_Corp_Panel_Sub_Type

@panel_sub_type_name varchar(50),
@panel_sub_type_descr varchar(250)

AS
insert into Corp_Panel_Sub_Type
(
 panel_sub_type_name,
 panel_sub_type_descr
)
values
(
 @panel_sub_type_name,
 @panel_sub_type_descr
)

Procedure Name: sp_Add_Corp_Panel_Test_Type

```
@panel_test_type_name varchar(50),  
@panel_test_type_descr varchar(250)
```

AS

insert into Corp_Panel_Test_Type

```
(  
    panel_test_type_name,  
    panel_test_type_descr
```

```
)  
values
```

```
(  
    @panel_test_type_name,  
    @panel_test_type_descr  
)
```

Procedure Name: sp_Add_Corp_Panel_Type

```
@panel_type_name varchar(50),  
@panel_type_descr varchar(250)  
  
AS  
insert into Corp_Panel_Type  
(  
    panel_type_name,  
    panel_type_descr  
)  
values  
(  
    @panel_type_name,  
    @panel_type_descr  
)
```

Procedure Name: sp_Add_Keys

```
@rec_id numeric(9,0),
@src_cat_name varchar(50),
@source_name varchar(250),
@src_indicator char(3),
@camp_name varchar(250),
@panel_name varchar(250)
```

AS

```
declare @product_id numeric(8,0),      --for keys table
@full_promo_key varchar(25),
@dt_valid_from smalldatetime,
@dt_valid_to smalldatetime,
@key_mail_qty numeric(8,0),
@list_cost_basis varchar(25),
@list_cat_name varchar(250),
@list_name varchar(250),
@list_segment_name varchar(250),      -- for info from panels table
@PANEL_TYPE varchar(50),
@PANEL_SUBTYPE varchar(50),
@TEST_TYPE varchar(50),
@PANEL_OFFER varchar(50),
@PANEL_OFFER_PRICE money,
@PANEL_PKG varchar(50),
@PANEL_PREMIUM varchar(50),
@PANEL_DATE_START smalldatetime,
@PANEL_DATE_END smalldatetime,
@GRP_NAME varchar(50),
@TITLE_NAME varchar(50),
@CAMP_TYPE varchar(50),
@CAMP_STATUS_NAME varchar(50),
@AGT_CODE varchar(50),
@AGT_DESCR varchar(50),
@CAMP_DATE_FR smalldatetime,
@CAMP_DATE_TO smalldatetime,
@E_SET_NAME varchar(50),
@R_SET_NAME varchar(50),
@newstand_sale_marker varchar(50),
@newstand_sale NUMERIC(18)
```

```
select @full_promo_key = full_promo_key,
@dt_valid_to = dt_valid_to,
@product_id = product_id,
@key_mail_qty = mail_qty
from avail_keys
where rec_id=@rec_id
```

```
--get panel and other info for key table
declare c_panel_info cursor for
select PANEL_TYPE ,
```

```
PANEL_SUBTYPE ,
TEST_TYPE ,
PANEL_OFFER,
PANEL_OFFER_PRICE ,
PANEL_PKG ,
PANEL_PREMIUM ,
PANEL_DATE_START,
PANEL_DATE_END ,
GRP_NAME ,
TITLE_NAME ,
CAMP_TYPE ,
CAMP_STATUS_NAME ,
AGT_CODE ,
AGT_DESCR ,
CAMP_DATE_FR ,
CAMP_DATE_TO, E_SET_NAME, R_SET_NAME
from panels where product_id = @product_id and src_cat_name = @src_cat_name and
source_name = @source_name
and camp_name = @camp_name and panel_name = @panel_name
select @PANEL_TYPE = PANEL_TYPE ,
@PANEL_SUBTYPE = PANEL_SUBTYPE ,
@TEST_TYPE = TEST_TYPE ,
@PANEL_OFFER = PANEL_OFFER,
@PANEL_OFFER_PRICE = PANEL_OFFER_PRICE ,
@PANEL_PKG = PANEL_PKG ,
@PANEL_PREMIUM = PANEL_PREMIUM ,
@PANEL_DATE_START = PANEL_DATE_START,
@PANEL_DATE_END = PANEL_DATE_END,
@GRP_NAME = GRP_NAME,
@TITLE_NAME = TITLE_NAME ,
@CAMP_TYPE = CAMP_TYPE ,
@CAMP_STATUS_NAME = CAMP_STATUS_NAME ,
@AGT_CODE = AGT_CODE ,
@AGT_DESCR = AGT_DESCR ,
@CAMP_DATE_FR = CAMP_DATE_FR ,
@CAMP_DATE_TO = CAMP_DATE_TO,
@E_SET_NAME = E_SET_NAME,
@R_SET_NAME = R_SET_NAME ,
@newstand_sale_marker = newstand_sale_marker,
@newstand_sale = Newstand_sale
from panels
where product_id = @product_id
and src_cat_name = @src_cat_name
and source_name = @source_name
and camp_name = @camp_name
and panel_name = @panel_name

insert into keys
(
    full_promo_key,

    dt_valid_from,
    dt_valid_to,
    src_cat_name,
    src_indicator,
```

panel_name,
camp_name,
product_id,
source_name,
key_mail_qty,

PANEL_TYPE,
PANEL_SUBTYPE,
TEST_TYPE,
PANEL_OFFER,
PANEL_OFFER_PRICE,
PANEL_PKG,
PANEL_PREMIUM,
PANEL_DATE_START,
PANEL_DATE_END,
GRP_NAME,
TITLE_NAME,
CAMP_TYPE,
CAMP_STATUS_NAME,
AGT_CODE,
AGT_DESCR,
CAMP_DATE_FR,
CAMP_DATE_TO,
E_SET_NAME,
R_SET_NAME,
newstand_sale_marker,
newstand_sale,
list_cost_basis,
list_cat_name,
list_name,
list_segment_name

)

values

(
 @full_promo_key,
 dateadd(month,-20,@dt_valid_to),
 @dt_valid_to,
 @src_cat_name,
 @src_indicator,
 @panel_name,
 @camp_name,
 @product_id,
 @source_name,
 @key_mail_qty,

 @PANEL_TYPE,
 @PANEL_SUBTYPE,
 @TEST_TYPE,
 @PANEL_OFFER,
 @PANEL_OFFER_PRICE,
 isNull(@PANEL_PKG,'Undefined'),
 @PANEL_PREMIUM,
 @PANEL_DATE_START,
 @PANEL_DATE_END,
 @GRP_NAME,
 @TITLE_NAME,

```
@CAMP_TYPE,  
@CAMP_STATUS_NAME,  
@AGT_CODE,  
@AGT_DESCR,  
@CAMP_DATE_FR,  
@CAMP_DATE_TO,  
'Undefined',  
'Undefined',  
isNull(@newstand_sale_marker,'N'),  
@newstand_sale,  
'Mail QTY',  
'Undefined',  
'Undefined',  
'Undefined'  
)  
  
if (@@error = 0)  
begin  
Delete from avail_keys where rec_id = @rec_id  
end  
close c_panel_info  
deallocate c_panel_info
```


Procedure Name: sp_Add_Keys2

```
@rec_id numeric(9,0),
@src_cat_name varchar(30),
@source_name varchar(250),
@src_indicator char(3),
@camp_name varchar(250),
@panel_name varchar(250)
```

AS

```
declare @product_id numeric(8,0),      --for keys table
@full_promo_key varchar(25),
@dt_valid_from smalldatetime,
@dt_valid_to smalldatetime,
@key_mail_qty numeric(8,0),
-- for info from panels table
@PANEL_TYPE varchar(50),
@PANEL_SUBTYPE varchar(50),
@TEST_TYPE varchar(50),
@PANEL_OFFER varchar(50),
@PANEL_OFFER_PRICE money,
@PANEL_PKG varchar(50),
@PANEL_PREMIUM varchar(50),
@PANEL_DATE_START smalldatetime,
@PANEL_DATE_END smalldatetime,
@GRP_NAME varchar(50),
@TITLE_NAME varchar(50),
@CAMP_TYPE varchar(50),
@CAMP_STATUS_NAME varchar(50),
@AGT_CODE varchar(50),
@AGT_DESCR varchar(50),
@CAMP_DATE_FR smalldatetime,
@CAMP_DATE_TO smalldatetime
```

```
declare c_avail_key cursor for
select full_promo_key, dt_valid_to, product_id, mail_qty from avail_keys where rec_id=@rec_id
```

```
open c_avail_key
fetch c_avail_key into @full_promo_key, @dt_valid_to, @product_id, @key_mail_qty
```

```
--get panel and other info for key table
declare c_panel_info cursor for
select PANEL_TYPE ,
PANEL_SUBTYPE ,
TEST_TYPE ,
PANEL_OFFER,
PANEL_OFFER_PRICE ,
PANEL_PKG ,
PANEL_PREMIUM ,
PANEL_DATE_START,
```

```
PANEL_DATE_END ,
GRP_NAME ,
TITLE_NAME ,
CAMP_TYPE ,
CAMP_STATUS_NAME ,
AGT_CODE ,
AGT_DESCR ,
CAMP_DATE_FR ,
CAMP_DATE_TO from panels where product_id = @product_id and src_cat_name =
@src_cat_name and source_name = @source_name
and camp_name = @camp_name and panel_name = @panel_name

open c_panel_info
fetch c_panel_info into
    @PANEL_TYPE,
    @PANEL_SUBTYPE,
    @TEST_TYPE,
    @PANEL_OFFER,
    @PANEL_OFFER_PRICE,
    @PANEL_PKG,
    @PANEL_PREMIUM,
    @PANEL_DATE_START,
    @PANEL_DATE_END,
    @GRP_NAME,
    @TITLE_NAME,
    @CAMP_TYPE,
    @CAMP_STATUS_NAME,
    @AGT_CODE,
    @AGT_DESCR,
    @CAMP_DATE_FR,
    @CAMP_DATE_TO

print dateadd(month,-20,@dt_valid_to)
insert into keys
(
    full_promo_key,

    dt_valid_from,
    dt_valid_to,
    src_cat_name,
    src_indicator,
    panel_name,
    camp_name,
    product_id,
    source_name,
    key_mail_qty,

    PANEL_TYPE,
    PANEL_SUBTYPE,
    TEST_TYPE,
    PANEL_OFFER,
    PANEL_OFFER_PRICE,
    PANEL_PKG,
    PANEL_PREMIUM,
    PANEL_DATE_START,
```



```
PANEL_DATE_END,  
GRP_NAME,  
TITLE_NAME,  
CAMP_TYPE,  
CAMP_STATUS_NAME,  
AGT_CODE,  
AGT_DESCR,  
CAMP_DATE_FR,  
CAMP_DATE_TO
```

```
)
```

```
values
```

```
( @full_promo_key,  
  dateadd(month,-20,@dt_valid_to),  
  @dt_valid_to,  
  @src_cat_name,  
  @src_indicator,  
  @panel_name,  
  @camp_name,  
  @product_id,  
  @source_name,  
  @key_mail_qty,  
  
  @PANEL_TYPE,  
  @PANEL_SUBTYPE,  
  @TEST_TYPE,  
  @PANEL_OFFER,  
  @PANEL_OFFER_PRICE,  
  @PANEL_PKG,  
  @PANEL_PREMIUM,  
  isNull( @PANEL_DATE_START,null) ,  
  isNull(@PANEL_DATE_END,null) ,  
  @GRP_NAME,  
  @TITLE_NAME,  
  @CAMP_TYPE,  
  @CAMP_STATUS_NAME,  
  @AGT_CODE,  
  @AGT_DESCR,  
  isNull(@CAMP_DATE_FR,null) ,  
  isNull( @CAMP_DATE_TO,null)
```

```
)
```

```
close c_avail_key  
deallocate c_avail_key
```

```
Delete from avail_keys where rec_id = @rec_id
```

```
close c_panel_info  
deallocate c_panel_info
```



Procedure Name: sp_Add_Panel

```
@panel_name varchar(250), --1
@panel_type varchar(50), --2
@panel_subtype varchar(50), --3
@test_type varchar(50), --4
@panel_pkg varchar(50), --5
@panel_date_start datetime, --6
@panel_date_end datetime, --7
@e_set_name varchar(50), --8
@r_set_name varchar(50), --9
@newstand_sale_marker char(1), --10
```

```
@product_id numeric(8,0), --11
@grp_name varchar(50), --12
@title_name varchar (250), --13
@src_cat_name varchar(30), --14
@source_name varchar (250), --15
@src_indicator char (3), --16
@camp_name varchar(250), --17
@camp_type varchar(50), --18
@camp_status_name varchar(20), --19
@agt_code varchar(10), --20
@agt_descr varchar(50), --21
@camp_date_fr datetime, --22
@camp_date_to datetime --23
```

AS
declare

```
@IE_SET_NAME varchar(50),
@IR_SET_NAME varchar(50),
@INewStand_sale NUMERIC(18)
--get expense and revenue set names for pass down if blank
```

```
select @IE_SET_NAME = E_SET_NAME,
@IR_SET_NAME = R_SET_NAME,
@INewStand_sale = Newstand_sale
from campaigns where product_id = @PRODUCT_ID
and SRC_CAT_NAME = @SRC_CAT_NAME and SOURCE_NAME = @SOURCE_NAME
and CAMP_NAME = @CAMP_NAME
```

insert into panels

```
(
    panel_name,
    panel_type,
    panel_subtype,
    test_type,
    panel_pkg,
    panel_date_start,
    panel_date_end,
    e_set_name,
```

r_set_name,
newstand_sale_marker,

product_id,
grp_name,
title_name,
src_cat_name,
source_name,
src_indicator,
camp_name,
camp_type,
camp_status_name,
agt_code,
agt_descr,
camp_date_fr,
camp_date_to,
Newstand_sale

)
values
(

@panel_name,
isnull(@panel_type,'Undefined'),
isnull(@panel_subtype,'Undefined'),
isnull(@test_type,'Undefined'),
isnull(@panel_pkg,'Undefined'),
@panel_date_start,
@panel_date_end,
isnull(@e_set_name,@IE_SET_NAME),
isnull(@r_set_name,@IR_SET_NAME),
@newstand_sale_marker,

@product_id,
@grp_name,
@title_name,
@src_cat_name,
@source_name,
@src_indicator,
@camp_name,
@camp_type,
@camp_status_name,
@agt_code,
@agt_descr,
@camp_date_fr,
@camp_date_to,
@lNewStand_sale

)



Procedure Name: sp_add_panel_def

```
@product_id int
, @panel_scheme varchar(50)
, @panel_codes varchar(250)
, @panel_name varchar(50)
```

AS

insert into autosetup_panel_def

```
(
    product_id
    , panel_scheme
    , panel_codes
    , panel_name
)
values
(
    @product_id
    , @panel_scheme
    , @panel_codes
    , @panel_name
)
```



Procedure Name: sp_Add_prior_Source_desc

```
@prior_source_identifier char (2)
, @prior_source_desc varchar (20)
, @billing_cur_src_desc varchar (255)
, @model_source_cat varchar(50)
, @model_source varchar(50)
, @agent_src_desc varchar(50)
, @valid_from_date datetime
, @valid_to_date datetime
, @product_id numeric(18, 0)
```

as

insert into Prior_source_desc

```
(
    product_id
    , prior_source_identifier
    , prior_source_desc
    , billing_cur_src_desc
    , model_source_cat
    , model_source
    , agent_src_desc
    , valid_from_date
    , valid_to_date
)
values (
    @product_id
    , @prior_source_identifier
    , @prior_source_desc
    , @billing_cur_src_desc
    , @model_source_cat
    , @model_source
    , @agent_src_desc
    , @valid_from_date
    , @valid_to_date
)
```



Procedure Name: sp_Add_Prod_Camp_Status

```
@camp_status_name varchar(20),  
@camp_status_descr varchar(250),  
  
@product_id numeric(8,0)
```

AS

insert into Prod_Camp_Status

```
(  
    camp_status_name,  
    camp_status_descr,  
    product_id  
)  
values  
(  
    @camp_status_name,  
    @camp_status_descr,  
    @product_id  
)
```




Procedure Name: sp_Add_Prod_Camp_Type

@camp_type_name varchar(250),

@camp_type_descr varchar(250),

@product_id numeric(8,0)

AS

insert into Prod_Camp_Type

(

camp_type_name,

camp_type_descr,

product_id

)

values

(

@camp_type_name,

@camp_type_descr,

@product_id

)

Procedure Name: sp_Add_Prod_E_Set

```
@product_id numeric(8, 0) ,
@e_set_name varchar (50) ,
@print_cost numeric(11, 3) ,
@print_cpm_cpu varchar (10) ,
@print_basis varchar (20) ,
@merge_purge numeric(11, 3) ,
@merge_cpm_cpu varchar (10) ,
@merge_basis varchar (20) ,
@ltr_cost numeric(11, 3) ,
@ltr_cpm_cpu varchar (10) ,
@ltr_basis varchar (20) ,
@postage_out_cost numeric(11, 3) ,
@postage_out_cpm_cpu varchar (10) ,
@postage_out_basis varchar (20) ,
@postage_in_cost numeric(11, 3) ,
@postage_in_cpm_cpu varchar (10) ,
@postage_in_basis varchar (20) ,
@premium_cost numeric(11, 3) ,
@premium_cpm_cpu varchar (10) ,
@premium_basis varchar (20) ,
@badpay_cost numeric(11, 3) ,
@badpay_cpm_cpu varchar (10) ,
@badpay_basis varchar (20) ,
@billing_cost numeric(11, 3) ,
@billing_cpm_cpu varchar (10) ,
@billing_basis varchar (20) ,
@subs_svc_costs numeric(11, 3) ,
@subs_svc_cpm_cpu varchar (10) ,
@subs_svc_basis varchar (20) ,
@exp_other_costs numeric(11, 3) ,
@exp_other_cpm_cpu varchar (10) ,
@exp_other_basis varchar (20)
```

AS

insert into expense_sets

(

```
product_id ,
e_set_name ,
print_cost ,
print_cpm_cpu ,
print_basis ,
merge_purge ,
merge_cpm_cpu ,
merge_basis ,
ltr_cost ,
ltr_cpm_cpu ,
ltr_basis ,
postage_out_cost ,
postage_out_cpm_cpu ,
postage_out_basis ,
postage_in_cost ,
postage_in_cpm_cpu ,
```

```
postage_in_basis ,
premium_cost,
premium_cpm_cpu,
premium_basis ,
badpay_cost ,
badpay_cpm_cpu ,
badpay_basis ,
billing_cost ,
billing_cpm_cpu ,
billing_basis ,
subs_svc_costs ,
subs_svc_cpm_cpu ,
subs_svc_basis ,
exp_other_costs ,
exp_other_cpm_cpu ,
exp_other_basis
)
values
(
    @product_id ,
    @e_set_name ,
    @print_cost ,
    @print_cpm_cpu ,
    @print_basis,
    @merge_purge,
    @merge_cpm_cpu,
    @merge_basis,
    @ltr_cost ,
    @ltr_cpm_cpu ,
    @ltr_basis ,
    @postage_out_cost ,
    @postage_out_cpm_cpu ,
    @postage_out_basis ,
    @postage_in_cost ,
    @postage_in_cpm_cpu ,
    @postage_in_basis ,
    @premium_cost,
    @premium_cpm_cpu,
    @premium_basis ,
    @badpay_cost ,
    @badpay_cpm_cpu ,
    @badpay_basis ,
    @billing_cost ,
    @billing_cpm_cpu ,
    @billing_basis ,
    @subs_svc_costs ,
    @subs_svc_cpm_cpu ,
    @subs_svc_basis ,
    @exp_other_costs ,
    @exp_other_cpm_cpu ,
    @exp_other_basis
)
```

Procedure Name: sp_Add_Prod_List_cat

-- added on 01/05/2000

@name varchar(250),
@descr varchar(250),
@product_id numeric(8,0)

insert into List_cats

(
list_cat_name,
list_cat_descr,
product_id

)
values

(
@name,
@descr,
@product_id
)

Procedure Name: sp_Add_Prod_List_Name

-- added on 01/05/2000

```
@name varchar(250),  
@descr varchar(250),  
@product_id numeric(8,0)
```

AS

insert into List_names

```
(  
    list_name,  
    list_name_description,  
    product_id
```

```
)  
values
```

```
(  
    @name,  
    @descr,  
    @product_id  
)
```

Procedure Name: sp_Add_Prod_List_Segment

– added on 01/05/2000

```
@name varchar(250),  
@descr varchar(250),  
@product_id numeric(8,0)
```

AS

insert into List_Segments

```
(  
    list_segment_name,  
    list_segment_description,  
    product_id  
)
```

values

```
(  
    @name,  
    @descr,  
    @product_id  
)
```

Procedure Name: sp_Add_Prod_Panel_Package

```
@package_name varchar(50),  
@package_description varchar(250),  
@Product_id numeric(9)
```

AS

```
insert into Prod_panel_Package
```

```
(  product_id,  
    package_name,  
    package_description
```

```
)
```

```
values
```

```
(  @product_id,  
    @package_name,  
    @package_description
```

```
)
```

Procedure Name: sp_Add_Prod_R_Set

```
@product_id numeric(8, 0),  
@r_set_name varchar (50) ,  
@subs_revenue numeric(11, 3) ,  
@subs_cpm_cpu varchar (10) ,  
@ad_revenue numeric(11, 3),  
@ad_cpm_cpu varchar (10),  
@listrental_revenue numeric(11, 3) ,  
@listrental_cpm_cpu varchar (10),  
@other_revenue numeric(11, 3),  
@other_cpm_cpu varchar (10)
```

AS

insert into revenue_sets

```
( product_id ,  
  r_set_name ,  
  subs_revenue ,  
  subs_cpm_cpu ,  
  ad_revenue ,  
  ad_cpm_cpu ,  
  listrental_revenue ,  
  listrental_cpm_cpu ,  
  other_revenue ,  
  other_cpm_cpu )
```

values

```
( @product_id ,  
  @r_set_name ,  
  @subs_revenue ,  
  @subs_cpm_cpu ,  
  @ad_revenue ,  
  @ad_cpm_cpu ,  
  @listrental_revenue ,  
  @listrental_cpm_cpu ,  
  @other_revenue ,  
  @other_cpm_cpu )
```

Procedure Name: sp_add_renewal_psdef

```
    @product_id int
    , @src_cat_name varchar(50)
    , @src_indicator varchar(50)
    , @prior_src_indicator varchar(250)
    , @source_name varchar (250)
    , @valid_from datetime
    , @valid_to datetime
```

AS

```
declare @title varchar(50)
```

```
declare @grp_name varchar(50)
```

```
select @title = title_name
      from corp_info
      where product_id = @product_id
```

```
insert into autosetup_renewal_psdef
```

```
(
    product_id
    , title
    , src_cat_name
    , source_indicator
    , prior_source_indicator
    , source_name
    , valid_from_date
    , valid_to_date
)
```

)

values

```
(
    @product_id
    , @title
    , @src_cat_name
    , @src_indicator
    , @prior_src_indicator
    , @source_name
    , @valid_from
    , @valid_to
)
```

Procedure Name: sp_Add_Source

```
@source_name varchar (250),  
@src_indicator char (3),  
  
@product_id numeric(8,0),  
@src_cat_name varchar(30),  
@grp_name varchar(50),  
@title_name varchar (250)
```

AS

insert into source

```
(  
    product_id,  
    src_cat_name,  
    source_name,  
    src_indicator,  
    grp_name,  
    title_name  
)  
values  
(  
    @product_id,  
    @src_cat_name,  
    @source_name,  
    @src_indicator,  
    @grp_name,  
    @title_name  
)
```

Procedure Name: sp_Add_Source_description

```
@u_source_name varchar(250),
@u_source_description varchar(250)
AS
insert into Source_descriptions
(
    u_source_name,
    u_source_description
)
values
(
    @u_source_name,
    @u_source_description
)
```

Procedure Name: sp_cascade_product_id

```
INSERT INTO list_cats
SELECT product_id, 'Undefined',
FROM corp_info
WHERE product_id not in (
  select product_id
  FROM list_cats
  WHERE list_cat_name = 'Undefined' )
```

```
INSERT INTO list_names
SELECT product_id, 'Undefined',
FROM corp_info
WHERE product_id not in (
  select product_id
  FROM list_names
  WHERE list_name = 'Undefined' )
```

```
INSERT INTO list_segments
SELECT product_id, 'Undefined',
FROM corp_info
WHERE product_id not in (
  select product_id
  FROM list_segments
  WHERE list_segment_name = 'Undefined' )
```

```
INSERT INTO expense_sets
SELECT product_id, 'Undefined',
  null, null, null, null,
  null, null, null, null,
  null, null, null, null,
  null, null, null, null,
  null, null, null, null,
  null, null, null, null,
  null, null, null, null,
  null, null, null, null,
  null
FROM corp_info
WHERE product_id not in (
  select product_id
  FROM expense_sets
  WHERE e_set_name = 'Undefined' )
```

```
INSERT INTO revenue_sets
SELECT product_id, 'Undefined',
  null, null, null, null,
  null, null, null, null
FROM corp_info
WHERE product_id not in (
  select product_id
  FROM revenue_sets
  WHERE r_set_name = 'Undefined' )
```

```
INSERT INTO prod_camp_status
SELECT product_id, 'Undefined',"
FROM corp_info
WHERE product_id not in (
  select product_id
  FROM prod_camp_status
  WHERE camp_status_name = 'Undefined' )
```

```
INSERT INTO prod_camp_type
SELECT product_id, 'Undefined',"
FROM corp_info
WHERE product_id not in (
  select product_id
  FROM prod_camp_type
  WHERE camp_type_name = 'Undefined' )
```

```
INSERT INTO prod_panel_package
SELECT product_id, 'Undefined',"
FROM corp_info
WHERE product_id not in (
  select product_id
  FROM prod_panel_package
  WHERE package_name = 'Undefined' )
```

Procedure Name: sp_DeleteCamp_Statues

@product_id numeric(9),
@camp_status_name varchar(20)

AS

delete from camp_statuses where product_id = @product_id and camp_status_name =
@camp_status_name

Procedure Name: sp_DeleteCamp_Types

@product_id numeric(9),
@camp_type varchar(50)

AS

delete from camp_types where product_id = @product_id and camp_type = @camp_type

Procedure Name: sp_DeletePanel_SubTypes

@product_id numeric(9),
@panel_subtype varchar(50)

AS

delete from p_sub_types where product_id = @product_id and panel_subtype = @panel_subtype

Procedure Name: sp_DeletePanel_TestTypes

@product_id numeric(9),
@test_type varchar(50)

AS

delete from p_test_types where product_id = @product_id and test_type = @test_type

Procedure Name: sp_DeletePanel_Types

@product_id numeric(9),
@panel_type varchar(50)

AS

delete from p_types where product_id = @product_id and panel_type = @panel_type



Procedure Name: sp_DeleteRPT_Def

@rpt_id numeric(9)
AS

delete from rpt_Defs where rpt_id = @rpt_id

Procedure Name: sp_edit_autosetup

```
@src_cat_name varchar(50)
, @src_indicator varchar(10)
, @source_name varchar(250)
, @valid_from_date datetime
, @valid_to_date datetime
, @setup_mask varchar(50)
, @e_set_name varchar(50)
, @login Varchar(50)
, @rec_id numeric(8,0)
, @isDelete numeric(1,0)
```

AS

If (@isDelete=1)

begin

delete from autosetup_scheme where rec_id = @rec_id

end

if (@isDelete=0)

begin

UPDATE autosetup_scheme

SET src_cat_name = @src_cat_name

, source_name = @source_name

, src_indicator = @src_indicator

, valid_from_date = @valid_from_date

, valid_to_date = @valid_to_date

, setup_mask = @setup_mask

, e_set_name = @e_set_name

WHERE rec_id = @rec_id

end

Procedure Name: sp_edit_autosetup_campaign_def

```
@campaign_code varchar(50)
, @campaign_description varchar(200)
, @src_cat_name varchar(250)
, @campaign_date datetime
, @panel_scheme varchar(50)
, @rec_id numeric(8,0)
, @isDelete numeric(1,0)
```

AS

```
If (@isDelete=1)
begin
    delete from autosetup_campaign_def where rec_id = @rec_id
end
if (@isDelete=0)
begin
    UPDATE autosetup_campaign_def
    SET campaign_code = @campaign_code
      , campaign_description = @campaign_description
      , src_cat_name = @src_cat_name
      , campaign_date = @campaign_date
      , panel_scheme = @panel_scheme
    WHERE rec_id = @rec_id
End
```

Procedure Name: sp_edit_autosetup_panel_def

```
@panel_scheme varchar(50)
, @panel_codes varchar(255)
, @panel_name varchar(50)
, @rec_id numeric(8,0)
, @isDelete numeric(1,0)
```

AS

```
If (@isDelete=1)
begin
    delete from autosetup_panel_def where rec_id = @rec_id
end
if (@isDelete=0)
begin
    UPDATE autosetup_panel_def
    SET panel_scheme = @panel_scheme
      , panel_codes = @panel_codes
      , panel_name = @panel_name
    WHERE rec_id = @rec_id
end
```

Procedure Name: procedure sp_edit_autosetup_renewal_psdef

```
@src_cat_name varchar(50)
, @source_name varchar(50)
, @source_indicator varchar(50)
, @prior_source_indicator varchar(255)
, @valid_from_date datetime
, @valid_to_date datetime
, @rec_id numeric(8,0)
, @isDelete numeric(1,0)
```

AS

```
If (@isDelete=1)
begin
    delete from autosetup_renewal_psdef where rec_id = @rec_id
end
if (@isDelete=0)
begin
    UPDATE autosetup_renewal_psdef
    SET src_cat_name = @src_cat_name
      , source_name = @source_name
      , source_indicator = @source_indicator
      , prior_source_indicator = @prior_source_indicator
      , valid_from_date = @valid_from_date
      , valid_to_date = @valid_to_date
    WHERE rec_id = @rec_id
end
```

Procedure Name: sp_Edit_Campaign

```

@camp_name varchar(250), -- 1
@camp_date_fr smalldatetime, -- 2
@camp_type varchar(50), -- 3
@camp_status_name varchar(20), -- 4
--@camp_date_to smalldatetime, -- 5
--@agt_code varchar(10), -- 6
--@agt_descr varchar(50), -- 7
@e_set_name varchar(50), -- 8
@r_set_name varchar(50), -- 9
--@newstand_sale numeric(18), -- 10
@budgeted_gross_pct numeric(18,3), -- 11
@budgeted_vol numeric(18,3), -- 12
@login VARCHAR(50), -- 13

```

```

@rec_id numeric(8,0),
@isDelete numeric(1,0)

```

AS

declare

```

@IPRODUCT_ID numeric(8,0),
@ICAMP_NAME varchar(50),
@ISOURCE_NAME varchar(250),
@IE_set_name Varchar(50),
@IR_set_name Varchar(50),
@lsrc_cat_name VARCHAR(30),
@message VARCHAR(255)

```

--Copy vars

```

declare @copycampname varchar(50)
declare @copies int

```

--get camp, panel and other info for deletions from key table

begin --sp

```

select @IPRODUCT_ID = product_id,
@ICAMP_NAME = camp_name,
@ISOURCE_NAME = source_name,
@lsrc_cat_name = src_cat_name ,
@IE_set_name = E_set_name,
@IR_set_name = r_set_name

```

from campaigns

where rec_id = @rec_id

if (@IsDelete=2) --then copy existing campaign and associated records into new record(s)

begin

set @copycampname = "Copy __ of " + @camp_name

--Get number of next copy

select @copies = (count(*) + 1)

from campaigns

where camp_name like @copycampname

if @copies < 10 set @copycampname = "Copy " + cast(@copies as varchar(1)) + " of "

else set @copycampname = "Copy " + cast(@copies as varchar(2)) + " of "


```
--Insert new copy of campaign
insert campaigns (src_indicator,
                  src_cat_name,
                  product_id,
                  camp_type,
                  camp_name,
                  camp_status_name,
                  --agt_code,
                  --agt_descr,
                  camp_date_fr,
                  --camp_date_to,
                  e_set_name,
                  r_set_name,
                  grp_name,
                  title_name,
                  source_name,
                  newstand_sale,
                  budgeted_gross_pct,
                  budgeted_vol,
                  campaign_code)

--values
select            src_indicator,
                  src_cat_name,
                  product_id,
                  camp_type,
                  @copycampname + @camp_name as campname,
                  camp_status_name,
                  --agt_code,
                  --agt_descr,
                  camp_date_fr,
                  --dateadd(month,1,camp_date_to) as camp_date_to,
                  e_set_name,
                  r_set_name,
                  grp_name,
                  title_name,
                  source_name,
                  newstand_sale,
                  budgeted_gross_pct,
                  budgeted_vol,
                  campaign_code
from campaigns
where rec_id = @rec_id

--Insert new copies of panels from the campaign we are copying
insert panels
select @copycampname + @camp_name as camp_name,
      src_indicator,
      panel_name,
      src_cat_name,
      product_id,
      panel_type,
      e_set_name,
      panel_subtype,
      r_set_name,
      test_type,
```

```
panel_offer,  
panel_offer_price,  
panel_pkg,  
panel_premium,  
panel_date_start,  
panel_date_end,  
grp_name,  
title_name,  
source_name,  
camp_type,  
camp_status_name,  
agt_code,  
agt_descr,  
camp_date_fr,  
camp_date_to,  
Newstand_Sale_Marker,  
Newstand_Sale,  
Budgeted_gross_pct,  
Budgeted_vol,  
campaign_code  
, panel_code  
, null as order_qty  
, null as dist_qty  
from panels  
where product_id = @lPRODUCT_ID  
and src_cat_name = @lsrc_cat_name  
and source_name = @lSOURCE_NAME  
and camp_name = @lCAMP_NAME
```

```
end --if  
If (@isDelete=1) --then delete campaign and associated records  
begin  
  
declare c_keys cursor for /* r101 recordset - the main/reference one */  
select product_id, full_promo_key  
from keys  
where product_id = @lproduct_id  
and camp_name = @lcamp_name  
and SOURCE_NAME = @lSOURCE_NAME  
and src_cat_name = @lsrc_cat_name  
  
declare  
    @lkeys_PRODUCT_ID numeric(8,0),  
    @lfull_promo_key varchar(50)  
  
open c_keys  
fetch c_keys into @lkeys_product_id, @lfull_promo_key  
  
while (@@fetch_status = 0)  
begin  
    exec doRecycleKey @lkeys_product_id, @lfull_promo_key  
    fetch c_keys into @lkeys_product_id, @lfull_promo_key  
end  
close c_keys  
deallocate c_keys
```

```
delete
from keys
where product_id = @IPRODUCT_ID
and src_cat_name = @lsrc_cat_name
and source_name = @ISOURCE_NAME
and camp_name = @lCAMP_NAME

delete
from panels
where product_id = @IPRODUCT_ID
and src_cat_name = @lsrc_cat_name
and source_name = @ISOURCE_NAME
and camp_name = @lCAMP_NAME

delete from campaigns where rec_id = @rec_id
set @message = 'Campaign: ' + @lCAMP_NAME + ' Source: ' + @ISOURCE_NAME + ' Src_cat: ' +
@lsrc_cat_name + ', delete applied to panel and keys'
exec sp_loginfio @lproduct_id, @login, 'Delete Campaign', @message
end
if (@IsDelete=0) --Update
begin
    update keys
    set camp_name = @camp_name,
        camp_type=isNull(@camp_type,'Undefined'),
        camp_status_name = isNull( @camp_status_name,'Undefined'),
        camp_date_fr=@camp_date_fr,
        --camp_date_to=@camp_date_to,
        --agt_code=@agt_code,
        --agt_descr=@agt_descr,
        --Newstand_sale = @newStand_sale,
        budgeted_vol = @budgeted_vol, -- 11
        budgeted_gross_pct = @budgeted_gross_pct
    where product_id = @IPRODUCT_ID
    and source_name = @ISOURCE_NAME
    and camp_name = @lCAMP_NAME
    and src_cat_name = @lsrc_cat_name

    update panels
    set camp_name = @camp_name,
        camp_type=isNull(@camp_type,'Undefined'),
        camp_status_name = isNull( @camp_status_name,'Undefined'),
        camp_date_fr=@camp_date_fr,
        --camp_date_to=@camp_date_to,
        --agt_code=@agt_code,
        --agt_descr=@agt_descr,
        --Newstand_sale = @newStand_sale,
        budgeted_vol = @budgeted_vol, -- 11
        budgeted_gross_pct = @budgeted_gross_pct
    where product_id = @IPRODUCT_ID
    and source_name = @ISOURCE_NAME
    and camp_name = @lCAMP_NAME
    and src_cat_name = @lsrc_cat_name
```

```

update campaigns
set camp_name = @camp_name,
  camp_type=isNull(@camp_type,'Undefined'),
  camp_status_name = isNull( @camp_status_name,'Undefined'),
  camp_date_fr=@camp_date_fr,
  --camp_date_to=@camp_date_to,
  --agt_code=@agt_code,
  --agt_descr=@agt_descr,
  e_set_name=isNull(@e_set_name,'Undefined'),
  r_set_name=isNull(@r_set_name,'Undefined'),
  --newstand_sale = @newstand_sale,
  budgeted_vol = @budgeted_vol, -- 11
  budgeted_gross_pct = @budgeted_gross_pct
where rec_id = @rec_id
if (@e_Set_name <> @le_Set_name )
BEGIN
  update panels
  set e_set_name = @e_Set_name
  WHERE product_id = @IPRODUCT_ID
  and source_name = @ISOURCE_NAME
  and camp_name = @ICAMP_NAME
  and e_set_name = @le_Set_name
  and src_cat_name = @lsrc_cat_name

  update keys
  set e_set_name = @e_Set_name
  WHERE product_id = @IPRODUCT_ID
  and source_name = @ISOURCE_NAME
  and camp_name = @ICAMP_NAME
  and e_set_name = @le_Set_name
and src_cat_name = @lsrc_cat_name
END
if (@r_Set_name <> @lr_set_name )
BEGIN
  update panels
  set r_set_name = @r_Set_name
  WHERE product_id = @IPRODUCT_ID
  and source_name = @ISOURCE_NAME
  and camp_name = @ICAMP_NAME
  and r_set_name = @lr_Set_name
  and src_cat_name = @lsrc_cat_name
  update keys
  set r_set_name = @r_Set_name
  WHERE product_id = @IPRODUCT_ID
  and source_name = @ISOURCE_NAME
  and camp_name = @ICAMP_NAME
  and r_set_name = @lr_Set_name
  and src_cat_name = @lsrc_cat_name
END
close c_camp_info2
deallocate c_camp_info2
if @lcamp_name <> @camp_name
BEGIN
  set @message = 'Old Campaign: ' + @ICAMP_NAME + ', New Campaign: ' +
@CAMP_NAME + ', Source: ' +

```



IMT Subscription Marketing Reporting System

```
        @ISOURCE_NAME + ', Src_cat: ' + @lsrc_cat_name + ', edit applied to panel and keys'
    END
    ELSE
    BEGIN
        set @message = 'Campaign: ' + @ICAMP_NAME + ', Source: ' +
        @ISOURCE_NAME + ', Src_cat: ' + @lsrc_cat_name + ', edit applied to panel and keys'
    END
    exec sp_loginfo @lproduct_id, @login, 'Edit Campaign', @message
end --if
end --sp
```

Procedure Name: sp_Edit_Corp_Panel_Sub_Type

```
@panel_sub_type_name varchar(50),  
@panel_sub_type_descr varchar(250),
```

```
@rec_id numeric(8,0),  
@isDelete numeric(1,0)
```

AS

```
declare @old_value varchar(50)  
select @old_value = panel_sub_type_name  
From corp_panel_sub_type  
where rec_id = @rec_id
```

```
If (@isDelete=1)  
begin
```

```
    exec cascade_update 'panel_subtype', @old_value, 'Undefined', "PANELS", -1  
    delete from Corp_Panel_Sub_Type where rec_id = @rec_id
```

```
end
```

```
else
```

```
begin
```

```
    if @old_value <> @panel_sub_type_name  
    begin
```

```
        exec cascade_update 'panel_subtype', @old_value, @panel_sub_type_name, "PANELS", -1  
    end
```

```
        update Corp_Panel_Sub_Type  
        set panel_sub_type_name = @panel_sub_type_name,  
        panel_sub_type_descr = @panel_sub_type_descr where rec_id = @rec_id
```

```
end
```

Procedure Name: sp_Edit_Corp_Panel_Test_Type

```
@panel_test_type_name varchar(50),  
@panel_test_type_descr varchar(250),  
@rec_id numeric(8,0),  
@isDelete numeric(1,0)
```

AS

```
declare @old_value varchar(50)  
select @old_value = panel_test_type_name  
From corp_panel_test_type  
where rec_id = @rec_id
```

```
If (@isDelete=1)  
begin
```

```
    exec cascade_update 'panel_testtype', @old_value, 'Undefined', 'PANELS', -1  
    delete from Corp_Panel_Test_Type where rec_id = @rec_id
```

```
end  
else  
begin
```

```
    if @old_value <> @panel_test_type_name  
    begin  
        exec cascade_update 'panel_testtype', @old_value, @panel_test_type_name, 'PANELS', -1  
    end  
    update Corp_Panel_Test_Type  
    set panel_test_type_name = @panel_test_type_name,  
    panel_test_type_descr = @panel_test_type_descr where rec_id = @rec_id
```

```
end
```



Procedure Name: sp_Edit_Corp_Panel_Type

```
@panel_type_name varchar(50),  
@panel_type_descr varchar(250),  
@rec_id numeric(8,0),  
@isDelete numeric(1,0)
```

AS

```
declare @old_value varchar(50)
```

```
select @old_value = panel_type_name  
From corp_panel_type  
where rec_id = @rec_id
```

```
If (@isDelete=1)
```

```
begin
```

```
    exec cascade_update 'panel_type', @old_value, 'Undefined', 'PANELS', -1
```

```
    delete from Corp_Panel_Type where rec_id = @rec_id
```

```
end
```

```
else
```

```
begin
```

```
if @old_value <> @panel_type_name
```

```
begin
```

```
    exec cascade_update 'panel_type', @old_value, @panel_type_name, 'PANELS', -1
```

```
--    exec corp_cascade_update 'panel_type', @old_value, @panel_type_name
```

```
end
```

```
update Corp_Panel_Type
```

```
    set panel_type_name = @panel_type_name,
```

```
    panel_type_descr = @panel_type_descr where rec_id = @rec_id
```

```
end
```

Procedure Name: sp_Edit_Key

```
@P_KEY_DESCR varchar(250),
@MAIL_QTY_OVERRIDE varchar(10),  --decimal(18,0),
@Newstand_Sale numeric(18,0),
@Newstand_Sale_Marker char(1),
@list_cat_name varchar(250),
@list_name varchar(250),
@list_segment_name varchar(250),
@LIST_COST_ACTUAL_CPM numeric(18,0),
@list_cost_basis varchar(25),
@list_cost_factor numeric(9,2),
@merge_purge_qty numeric(9),
@E_SET_NAME varchar(50),
@R_SET_NAME varchar(50),
@panel_pkg varchar(50),
@subtotal_flag varchar(6),
@optional_effort_id Varchar(50),
@rec_id numeric(8,0),
@isDelete numeric(1,0)
```

AS

```
If (@isDelete=1)
begin
```

declare

```
    @rPRODUCT_ID numeric(8,2),
    @rFULL_PROMO_KEY varchar(50)
--get pid and key for recycle key routine
declare c_key_info_r cursor for
select product_id, full_promo_key
from keys where rec_id = @rec_id
open c_key_info_r
fetch c_key_info_r into
    @rPRODUCT_ID,
    @rFULL_PROMO_KEY
close c_key_info_r
deallocate c_key_info_r
```

```
execute doRecycleKey @rPRODUCT_ID, @rFULL_PROMO_KEY
delete from keys where rec_id = @rec_id
end
else
begin
```

--validate data

```
If not(@MAIL_QTY_OVERRIDE > 0)
    select @MAIL_QTY_OVERRIDE=null
If (@E_SET_NAME = "0")
    select @E_SET_NAME=null
If (@R_SET_NAME = "0")
    select @R_SET_NAME=null
--end data validation
```



update keys

```
set P_KEY_DESCR=@P_KEY_DESCR,  
MAIL_QTY_OVERRIDE=@MAIL_QTY_OVERRIDE,  
Newstand_Sale = @Newstand_Sale,  
Newstand_Sale_Marker = @Newstand_Sale_Marker,  
LIST_CAT_NAME=isnull(@list_cat_name,'Undefined'),  
LIST_NAME=isnull(@list_name,'Undefined'),  
LIST_SEGMENT_NAME=isnull(@list_segment_name,'Undefined'),  
LIST_COST_ACTUAL_CPM=@LIST_COST_ACTUAL_CPM,  
LIST_COST_BASIS = @list_cost_basis,  
list_cost_factor = @list_cost_factor,  
merge_purge_qty = @merge_purge_qty,  
e_set_name=isnull(@e_set_name,'Undefined'),  
r_set_name=isnull(@r_set_name,'Undefined'),  
PANEL_PKG=isnull(@panel_pkg,'Undefined'),  
subtotal_flag=@subtotal_flag,  
optional_effort_id = @optional_effort_id,  
date_assigned = getdate()  
where rec_id = @rec_id
```

end



Procedure Name: sp_Edit_Panel

```
@panel_name varchar(250), -- 0
@panel_date_start datetime, -- 1
@panel_type varchar(50), -- 2
@panel_subtype varchar(50), -- 3
@test_type varchar(50), -- 4
@panel_pkg varchar(50), -- 5
@e_set_name varchar(50), -- 6
@r_set_name varchar(50), -- 7
--@newstand_sale_marker char(1), -- 8
@login Varchar(50), --9

@rec_id numeric(8,0), -- 1
@isDelete int -- 1
```

AS

declare

```
@lPRODUCT_ID numeric(8,0),
@lPANEL_NAME varchar(50),
@lCAMP_NAME varchar(50),
@lSOURCE_NAME varchar(250),
@lSRC_CAT_NAME varchar(30),
@lFull_promo_key varchar(25),
@lkeys_product_id numeric(8,0),
@lE_SET_NAME varchar(50),
@lR_SET_NAME varchar(50),
@lpanel_pkg varchar(50),
@message Varchar(255),
@src_indicator char(3),
@grp_name varchar(50),
@copies int,
@copypanelname varchar(50)
```

--get panel and other info for deletions from key table

```
select @lPRODUCT_ID = PRODUCT_ID,
       @lPANEL_NAME = PANEL_NAME,
       @lCAMP_NAME = CAMP_NAME,
       @lSOURCE_NAME = SOURCE_NAME,
       @lSRC_CAT_NAME = SRC_CAT_NAME,
       @le_set_name = e_set_name,
       @lr_set_name = r_set_name,
       @lpanel_pkg = panel_pkg,
       @src_indicator = src_indicator,
       @grp_name = grp_name
```

from panels

where rec_id = @rec_id

If (@isDelete=1)

begin

declare c_keys cursor for /* r101 recordset - the main/reference one */

```
select product_id, full_promo_key
from keys
where product_id = @lproduct_id
  and camp_name = @lcamp_name
  and SOURCE_NAME = @lSOURCE_NAME
  and panel_name = @lpanel_name
  and src_cat_name = @lsrc_cat_name

open c_keys
fetch c_keys into @lkeys_product_id, @lfull_promo_key

while (@@fetch_status = 0)
  begin
    exec doRecycleKey @lkeys_product_id, @lfull_promo_key
    fetch c_keys into @lkeys_product_id, @lfull_promo_key
  end
close c_keys
deallocate c_keys

delete from keys
where product_id = @lPRODUCT_ID
  and src_cat_name = @lSRC_CAT_NAME
  and source_name = @lSOURCE_NAME
  and camp_name = @lCamp_name
  and panel_name=@lPANEL_NAME

delete from panels
where rec_id = @rec_id

--close c_panel_info2
--deallocate c_panel_info2
set @message = 'Panel:' + @lPANEL_NAME + ',Campaign:' + @lCAMP_NAME + ',Source:' +
@lSOURCE_NAME + ',Src_cat:' + @lsrc_cat_name + ',delete applied to panel and keys'
exec sp_loginfo @lproduct_id, @login, 'Delete Panel', @message

end

if (@lIsDelete=0)
begin
--declare
--get expense and revenue set names for pass down if blank
-- NOT USING ANYMORE, this stuff should be defined when keys are assigned
--      select @lE_Set_name = E_SET_NAME,
--      @lR_Set_name = R_SET_NAME
-- from campaigns where product_id = @lPRODUCT_ID
--and SRC_CAT_NAME = @lSRC_CAT_NAME and SOURCE_NAME = @lSOURCE_NAME and
CAMP_NAME = @lCAMP_NAME

PRINT 'e set name--' + @e_set_name + '--e set name --' + @lE_SET_NAME + '--' + CAST(
@lPRODUCT_ID as varchar)

update keys
set panel_name = @panel_name,
panel_type=isnull(@panel_type,'Undefined'),
panel_subtype=isnull(@panel_subtype,'Undefined'),
```

```

test_type=isnull(@test_type,'Undefined'),
-- panel_pkg=isnull(@panel_pkg,'Undefined'),
panel_date_start=@panel_date_start
-- newstand_sale_marker = @newstand_sale_marker
-- e_set_name=isnull(@e_set_name,@lE_SET_NAME),
-- r_set_name=isnull(@r_set_name,@lR_SET_NAME)
where (panel_name + ':' + camp_name + ':' + src_cat_name + ':' + source_name + ':' +
cast(product_id as varchar) )
= (select panel_name + ':' + camp_name + ':' + src_cat_name + ':' + source_name + ':' +
cast(product_id as varchar) from panels where rec_id = @rec_id )
if (@le_set_name <> @e_set_name)
begin
update keys
set e_set_name = @e_set_name
-- panel_type=isnull(@panel_type,'Undefined'),
-- panel_subtype=isnull(@panel_subtype,'Undefined'),
-- test_type=isnull(@test_type,'Undefined'),
-- panel_offer=@panel_offer,
-- panel_offer_price=@panel_offer_price,
-- panel_pkg=isnull(@panel_pkg,'Undefined'),
-- panel_premium=@panel_premium,
-- panel_date_start=@panel_date_start,
-- panel_date_end=@panel_date_end

-- r_set_name=isnull(@r_set_name,@lR_SET_NAME)
where e_set_name = @le_set_name and
(panel_name + ':' + camp_name + ':' + src_cat_name + ':' + cast(product_id as varchar) )
= (select panel_name + ':' + camp_name + ':' + src_cat_name + ':' + cast(product_id as
varchar)
from panels where rec_id = @rec_id )
end
if (@lr_set_name <> @r_set_name)
begin
update keys
set r_set_name = @r_set_name
-- panel_type=isnull(@panel_type,'Undefined'),
-- panel_subtype=isnull(@panel_subtype,'Undefined'),
-- test_type=isnull(@test_type,'Undefined'),
-- panel_offer=@panel_offer,
-- panel_offer_price=@panel_offer_price,
-- panel_pkg=isnull(@panel_pkg,'Undefined'),
-- panel_premium=@panel_premium,
-- panel_date_start=@panel_date_start,
-- panel_date_end=@panel_date_end

-- r_set_name=isnull(@r_set_name,@lR_SET_NAME)
where r_set_name = @lr_set_name and
(panel_name + ':' + camp_name + ':' + src_cat_name + ':' +
source_name + ':' + cast(product_id as varchar) ) =
(select panel_name + ':' + camp_name + ':' + src_cat_name + ':' +
source_name + ':' + cast(product_id as varchar)
from panels where rec_id = @rec_id )
end
if (@lPanel_pkg <> @panel_pkg)
begin

```



```
update keys
    set Panel_Pkg = @panel_pkg
-- panel_type=isnull(@panel_type,'Undefined'),
-- panel_subtype=isnull(@panel_subtype,'Undefined'),
-- test_type=isnull(@test_type,'Undefined'),
-- panel_offer=@panel_offer,
-- panel_offer_price=@panel_offer_price,
-- panel_pkg=isnull(@panel_pkg,'Undefined'),
-- panel_premium=@panel_premium,
-- panel_date_start=@panel_date_start,
-- panel_date_end=@panel_date_end

-- r_set_name=isnull(@r_set_name,@IR_SET_NAME)
-- where Panel_Pkg = @lpanel_pkg and
--     (panel_name + ':' + camp_name + ':' + src_cat_name + ':' +
--      source_name + ':' + cast(product_id as varchar) ) =
--      (select panel_name + ':' + camp_name + ':' + src_cat_name + ':' +
--       source_name + ':' + cast(product_id as varchar)
--      from panels where rec_id = @rec_id )
end

update panels
    set panel_name = @panel_name,
    panel_type=isnull(@panel_type,'Undefined'),
    panel_subtype=isnull(@panel_subtype,'Undefined'),
    test_type=isnull(@test_type,'Undefined'),
    panel_pkg=isnull(@panel_pkg,'Undefined'),
    panel_date_start=@panel_date_start,
    e_set_name=isnull(@e_set_name,@IE_SET_NAME),
    r_set_name=isnull(@r_set_name,@IR_SET_NAME)
-- newstand_sale_marker = @newstand_sale_marker
-- where rec_id = @rec_id

if @lpanel_name <> @panel_name
BEGIN
    set @message = 'Old Panel:' + @IPANEL_NAME + ',New Panel:' + @Panel_name +
        ',Campaign:' + @ICAMP_NAME + ',Source:' + @ISOURCE_NAME + ',Src_cat:' +
        @lsrc_cat_name + ',edit applied to panel and keys'

END
ELSE
BEGIN

    set @message = 'Panel:' + @PANEL_NAME +
        ',Campaign:' + @ICAMP_NAME + ',Source:' + @ISOURCE_NAME + ',Src_cat:' +
        @lsrc_cat_name + ',edit applied to panel and keys'
end
exec sp_loginfo @lproduct_id, @login, 'Edit Panel', @message

end

if (@IsDelete=2) --then copy existing panel
```

begin

```
set @copypanelname = 'Copy __ of ' + @panel_name
--Get number of next copy
select @copies = (count(*) + 1)
      from panels
      where panel_name like @copypanelname
      and camp_name = @lCamp_name
      and src_cat_name = @lsrc_cat_name
      and product_id = @lproduct_id
      and source_name = @lsource_name
if @copies < 10 set @copypanelname = "Copy " + cast(@copies as varchar(1)) + " of "
else           set @copypanelname = "Copy " + cast(@copies as varchar(2)) + " of "
set @panel_name = @copypanelname + @panel_name
insert into panels (panel_name
                  , panel_type
                  , panel_subtype
                  , test_type
                  , panel_pkg
                  , panel_date_start
                  , e_set_name
                  , r_set_name
                  , product_id
                  , camp_name
                  , source_name
                  , src_cat_name
                  , src_indicator
                  , grp_name)
values
  (@panel_name
   , isnull(@panel_type,'Undefined')
   , isnull(@panel_subtype,'Undefined')
   , isnull(@test_type,'Undefined')
   , isnull(@panel_pkg,'Undefined')
   , @panel_date_start
   , isnull(@e_set_name,@IE_SET_NAME)
   , isnull(@r_set_name,@IR_SET_NAME)
   , @IPRODUCT_ID
   , @ICAMP_NAME
   , @ISOURCE_NAME
   , @ISRC_CAT_NAME
   , @src_indicator
   , @grp_name
  )
```

end

Procedure Name: sp_edit_prior_Source_desc

```
@prior_source_identifier char (2)
, @prior_source_desc varchar (20)
, @billing_cur_src_desc VARCHAR(255)
, @model_source_cat varchar(50)
, @model_source varchar(50)
, @agent_src_desc varchar(50)
, @valid_from_date datetime
, @valid_to_date datetime
, @rec_id numeric(8,0)
, @isDelete int
AS
declare @old_value varchar(50), @old_product_id numeric(9)

select @old_value = prior_source_identifier,
       @old_product_id = product_id
  From prior_source_desc
 where rec_id = @rec_id

If (@isDelete=1)
  begin
    exec cascade_update 'prior_source_identifier', @old_value, '-', 'KEYS', @old_product_id
    delete from prior_source_desc where rec_id = @rec_id
  end
else
  begin
    exec cascade_update 'prior_source_identifier', @old_value, '-', 'Est_max_effort',
    @old_product_id
    update prior_source_desc
      set --prior_source_identifier = @prior_source_identifier ,
          prior_source_desc = @prior_source_desc
        , billing_cur_src_desc = @billing_cur_src_desc
        , valid_from_date = @valid_from_date
        , valid_to_date = @valid_to_date
        , agent_src_desc = @agent_src_desc
        , model_source = @model_source
        , model_source_cat = @model_source_cat
      where rec_id = @rec_id
  end
end
```

Procedure Name: sp_Edit_Prod_Camp_Status

@camp_status_name varchar(20),
@camp_status_descr varchar(250),

@rec_id numeric(8,0),
@isDelete numeric(1,0)

AS

Declare @old_value VARCHAR(255), @product_id numeric(9)

```
select @old_value =  
    camp_status_name,  
    @product_id = product_id  
FROM prod_camp_status  
WHERE rec_id = @rec_id
```

If (@isDelete=1)
begin

```
    exec cascade_update 'camp_status_name', @old_value, 'Undefined', "CAMPAIGNS", @product_id  
    delete from Prod_Camp_Status where rec_id = @rec_id
```

end
else

begin

```
    if @old_value <> @camp_status_name  
    begin
```

```
        exec cascade_update 'camp_status_name', @old_value, @camp_status_name, 'CAMPAIGNS',
```

```
@product_id
```

```
--        exec prod_cascade_update_w_camp 'camp_status_name', @old_value, @camp_status_name ,  
@product_id  
    end
```

update Prod_Camp_Status

```
    set camp_status_name = @camp_status_name,
```

```
    camp_status_descr = @camp_status_descr where rec_id = @rec_id
```

end



Procedure Name: sp_Edit_Prod_Camp_Type

```
@camp_type_name varchar(20),  
@camp_type_descr varchar(250),  
@rec_id numeric(8,0),  
@isDelete numeric(1,0)
```

AS

Declare @old_value VARCHAR(255), @product_id numeric(9)

```
select @old_value =  
    camp_type_name,  
    @product_id = product_id  
FROM prod_camp_type  
WHERE rec_id = @rec_id
```

If (@isDelete=1)

begin

```
exec cascade_update 'camp_type', @old_value, 'Undefined', 'CAMPAIGNS', @product_id  
-- exec prod_cascade_update_w_camp 'camp_type', @old_value, 'Undefined', @product_id
```

```
delete from Prod_Camp_Type where rec_id = @rec_id
```

end

else

begin

```
if @old_value <> @camp_type_name  
begin
```

```
    exec cascade_update 'camp_type_name', @old_value, @camp_type_name, 'CAMPAIGNS',  
    @product_id
```

```
--    exec prod_cascade_update_w_camp 'camp_type', @old_value, @camp_type_name, @product_id  
end
```

update Prod_Camp_Type

```
set camp_type_name = @camp_type_name,  
camp_type_descr = @camp_type_descr where rec_id = @rec_id
```

end



Procedure Name: sp_Edit_Prod_E_Set

```
@product_id numeric(8,0) ,
@e_set_name varchar(50) ,
@print_cost numeric(11,3) ,
@print_cpm_cpu varchar(10) ,
@print_basis varchar(20) ,
@merge_purge numeric(11,3) ,
@merge_cpm_cpu varchar(10) ,
@merge_basis varchar(20) ,
@ltr_cost numeric(11,3) ,
@ltr_cpm_cpu varchar(10) ,
@ltr_basis varchar(20) ,
@postage_out_cost numeric(11,3) ,
@postage_out_cpm_cpu varchar(10) ,
@postage_out_basis varchar(20) ,
@postage_in_cost numeric(11,3) ,
@postage_in_cpm_cpu varchar(10) ,
@postage_in_basis varchar(20) ,
@premium_cost numeric(11,3) ,
@premium_cpm_cpu varchar(10) ,
@premium_basis varchar(20) ,
@badpay_cost numeric(11,3) ,
@badpay_cpm_cpu varchar(10) ,
@badpay_basis varchar(20) ,
@billing_cost numeric(11,3) ,
@billing_cpm_cpu varchar(10) ,
@billing_basis varchar(20) ,
@subs_svc_costs numeric(11,3) ,
@subs_svc_cpm_cpu varchar(10) ,
@subs_svc_basis varchar(20) ,
@exp_other_costs numeric(11,3) ,
@exp_other_cpm_cpu varchar(10) ,
@exp_other_basis varchar(20) ,
@rec_id numeric(8,0),
@isDelete numeric(1,0)
```

AS

```
Declare @old_value VARCHAR(255), @my_product_id numeric(9)
declare @e_set_id numeric(9)
```

```
select @old_value =
    e_set_name,
    @my_product_id = product_id
FROM expense_sets
WHERE rec_id = @rec_id
```

If (@isDelete=1)

begin

```
exec cascade_update 'e_set_name', @old_value, 'Undefined','CAMPAIGNS', @product_id
```

```
delete from Expense_sets where rec_id = @rec_id
```

end

```

else
begin
  if @old_value <> @e_set_name
  begin
    set @e_set_id = -1
    select @e_set_id = rec_id
      from expense_sets
     where e_set_name = @e_set_name
        and product_id = @product_id

    if @e_set_id = -1 exec sp_Add_Prod_E_Set --Create New Record, set everything to new e set and
delete old one

    @product_id, @e_set_name, @print_cost, @print_cpm_cpu, @print_basis,
@merge_purge, @merge_cpm_cpu, @merge_basis
    , @ltr_cost, @ltr_cpm_cpu, @ltr_basis, @postage_out_cost, @postage_out_cpm_cpu,
@postage_out_basis, @postage_in_cost, @postage_in_cpm_cpu
    , @postage_in_basis, @premium_cost, @premium_cpm_cpu, @premium_basis,
@badpay_cost, @badpay_cpm_cpu, @badpay_basis, @billing_cost
    , @billing_cpm_cpu, @billing_basis, @subs_svc_costs, @subs_svc_cpm_cpu,
@subs_svc_basis, @exp_other_costs, @exp_other_cpm_cpu
    , @exp_other_basis

    exec cascade_update 'e_set_name', @old_value, @e_set_name, 'CAMPAIGNS', @product_id
delete from expense_sets where rec_id = @rec_id

  end
else
  update expense_sets
    set e_set_name = @e_set_name ,
        print_cost = @print_cost ,
        print_cpm_cpu = @print_cpm_cpu ,
        print_basis = @print_basis ,
        merge_purge = @merge_purge ,
        merge_cpm_cpu = @merge_cpm_cpu ,
        merge_basis = @merge_basis ,
        ltr_cost = @ltr_cost ,
        ltr_cpm_cpu = @ltr_cpm_cpu ,
        ltr_basis = @ltr_basis ,
        postage_out_cost = @postage_out_cost ,
        postage_out_cpm_cpu = @postage_out_cpm_cpu ,
        postage_out_basis = @postage_out_basis ,
        postage_in_cost = @postage_in_cost ,
        postage_in_cpm_cpu = @postage_in_cpm_cpu ,
        postage_in_basis = @postage_in_basis ,
        premium_cost = @premium_cost ,
        premium_cpm_cpu = @premium_cpm_cpu ,
        premium_basis = @premium_basis ,
        badpay_cost = @badpay_cost ,
        badpay_cpm_cpu = @badpay_cpm_cpu ,
        badpay_basis = @badpay_basis ,
        billing_cost = @billing_cost ,
        billing_cpm_cpu = @billing_cpm_cpu ,
        billing_basis = @billing_basis ,
        subs_svc_costs = @subs_svc_costs ,
        subs_svc_cpm_cpu = @subs_svc_cpm_cpu ,
        subs_svc_basis = @subs_svc_basis ,

```

```
exp_other_costs = @exp_other_costs ,  
exp_other_cpm_cpu = @exp_other_cpm_cpu ,  
exp_other_basis = @exp_other_basis  
where rec_id = @rec_id  
end
```



Procedure Name: sp_Edit_Prod_List_Cat

```
@name varchar(250),
@descr varchar(250),
@rec_id numeric(8,0),
@isDelete numeric(1,0)
AS
Declare @old_value VARCHAR(255),
        @product_id numeric(9)

select @old_value =
        list_cat_name,
        @product_id = product_id
FROM list_cats
WHERE rec_id = @rec_id

If (@isDelete=1)
begin
    exec cascade_update 'LIST_CAT_NAME', @old_value, 'Undefined', "KEYS", @product_id
    delete from List_Cats where rec_id = @rec_id
end
else
begin
    if @name <> @old_value
    begin
        exec cascade_update 'LIST_CAT_NAME', @old_value, @name, "KEYS", @product_id
    end
end

update List_Cats
    set List_Cat_name = @name,
        List_Cat_descr = @descr where rec_id = @rec_id
end
```



Procedure Name: sp_Edit_Prod_List_Name

```
@name varchar(250),
@descr varchar(250),
@rec_id numeric(8,0),
@isDelete numeric(1,0)
AS
Declare @old_value VARCHAR(255), @product_id numeric(9)

select @old_value =
    list_name,
    @product_id = product_id
FROM list_names
WHERE rec_id = @rec_id

If (@isDelete=1)
begin
    exec cascade_update 'LIST_NAME', @old_value, 'Undefined', "KEYS", @product_id
    delete from List_Names where rec_id = @rec_id
end
else
begin
    if @name <> @old_value
    begin
        exec cascade_update 'LIST_NAME', @old_value, @name, "KEYS", @product_id
    end
    update List_NAMES
    set List_name = @name,
        List_Name_description = @descr
    where rec_id = @rec_id
end
```



Procedure Name: sp_Edit_Prod_List_Segment

```
@name varchar(250),
@descr varchar(250),
@rec_id numeric(8,0),
@isDelete numeric(1,0)
AS

Declare @old_value VARCHAR(255),
        @product_id numeric(9)

select @old_value =
        list_segment_name,
        @product_id = product_id
FROM list_segments
WHERE rec_id = @rec_id

If (@isDelete=1)
begin
    exec cascade_update 'LIST_SEGMENT_NAME', @old_value, 'Undefined', "KEYS", @product_id
    delete from List_Segments where rec_id = @rec_id
end
else
begin

    if @name <> @old_value
    begin
        exec cascade_update 'LIST_SEGMENT_NAME', @old_value, @name, "KEYS", @product_id
    end

    update List_Segments
        set List_Segment_name = @name,
            List_Segment_description = @descr where rec_id = @rec_id
end
```


Procedure Name: sp_Edit_Prod_Panel_Package

```
@package_name varchar(50),  
@package_description varchar(250),  
@rec_id numeric(8,0),  
@isDelete numeric(1,0)
```

AS

```
If (@isDelete=1)  
begin  
delete from prod_panel_package where rec_id = @rec_id  
end  
else  
begin  
update prod_panel_package  
set package_name = @package_name,  
package_description = @package_description where rec_id = @rec_id  
end
```



Procedure Name: sp_edit_Prod_R_Set

```
@product_id numeric(8, 0),
@r_set_name varchar (50) ,
@subs_revenue numeric(11, 3) ,
@subs_cpm_cpu varchar (10) ,
@ad_revenue numeric(11, 3),
@ad_cpm_cpu varchar (10),
@listrental_revenue numeric(11, 3) ,
@listrental_cpm_cpu varchar (10),
@other_revenue numeric(11, 3),
@other_cpm_cpu varchar (10),
@rec_id numeric(8,0),
@isDelete numeric(1,0)
```

AS

Declare @old_value VARCHAR(255), @my_product_id numeric(9)

```
select @old_value = r_set_name,
       @my_product_id = product_id
FROM revenue_sets
WHERE rec_id = @rec_id
```

If (@isDelete=1)

begin

exec cascade_update 'r_set_name', @old_value, 'Undefined','CAMPAIGNS', @product_id

delete from Revenue_sets where rec_id = @rec_id

end

else

begin

if @old_value <> @r_set_name

begin

exec cascade_update 'r_set_name', @old_value, @r_set_name, 'CAMPAIGNS', @product_id

end

update revenue_sets

set product_id = @product_id ,

r_set_name = @r_set_name ,

subs_revenue = @subs_revenue ,

subs_cpm_cpu = @subs_cpm_cpu ,

ad_revenue = @ad_revenue,

ad_cpm_cpu = @ad_cpm_cpu,

listrental_revenue= @listrental_revenue,

listrental_cpm_cpu = @listrental_cpm_cpu ,

other_revenue = @other_revenue,

other_cpm_cpu = @other_cpm_cpu

where rec_id = @rec_id

end

Procedure Name: sp_Edit_Source

```
@source_name varchar(250),  
@src_indicator varchar(10),  
@login Varchar(50),  
@rec_id numeric(8,0),  
@isDelete numeric(1,0)
```

AS

If (@isDelete=1)

begin

delete from source where rec_id = @rec_id

end

else

begin

declare

```
@PRODUCT_ID numeric(8,0),  
@src_cat_name varchar(50),  
@old_source_name varchar(30),  
@old_src_ind varchar(3),  
@message VARCHAR(200)
```

```
SELECT @PRODUCT_ID = Product_id ,  
@src_cat_name = src_cat_name,  
@old_source_name = source_name ,  
@old_src_ind = src_indicator
```

FROM source where rec_id =@rec_id

IF @source_name <> @old_source_name or @src_indicator <> @old_src_ind

BEGIN

set @message = 'Change applied to Panel, Campaign and Key levels as well as Source'

UPDATE campaigns

SET source_name = @source_name,

src_indicator = @src_indicator

WHERE Product_id = @Product_id

and src_cat_name = @src_cat_name

and source_name = @old_source_name

and src_indicator = @old_src_ind

UPDATE Panels

SET source_name = @source_name,

src_indicator = @src_indicator

WHERE Product_id = @Product_id

and src_cat_name = @src_cat_name

and source_name = @old_source_name

and src_indicator = @old_src_ind

UPDATE keys

SET source_name = @source_name,

src_indicator = @src_indicator

WHERE Product_id = @Product_id

and src_cat_name = @src_cat_name

```
and source_name = @old_source_name  
and src_indicator = @old_src_ind
```

```
update source
```

```
  set source_name = @source_name,  
  src_indicator = @src_indicator where rec_id = @rec_id
```

```
exec sp_loginfio @product_id, @login, 'Edit Source', @message  
end  
end
```

Procedure Name: sp_edit_Source_description

```
@u_source_name varchar(250),  
@u_source_description varchar(250),  
@rec_id numeric(8,0),  
@isDelete numeric(1,0)
```

AS

If (@isDelete=1)

begin

delete from Source_descriptions where rec_id = @rec_id

end

else

begin

update Source_descriptions

set u_source_name = @u_source_name,

u_source_description = @u_source_description where rec_id = @rec_id

end

Procedure Name: sp_LogInfo

```
@product_id varchar(250),  
@login varchar(50),  
--log_timestamp  
@action varchar(50),  
@action_description varchar(100)
```

As

```
insert into logs (product_id,login,log_timestamp,action,action_description) values  
(@product_id,@login,getdate(),@action,@action_description)
```

Procedure Name: sp_RemoveDuplicate101Keys

@pid numeric(9)

AS

declare @r101_key varchar(10), @select_date smalldatetime, @product_id numeric(8,0), @rct_rec_num decimal(9), @r101_reorg_date smalldatetime, @keyCount numeric(8,0)

declare c_r101dup cursor for

select product_id, r101_key, r101_reorg_date, count(r101_key) as cnt, max(rct_rec_num) as maxRecNum
from r101_flow where product_id = @pid group by product_id, r101_key, r101_reorg_date order by
product_id, cnt desc, r101_key

open c_r101dup

fetch c_r101dup into @product_id, @r101_key, @r101_reorg_date, @keyCount, @rct_rec_num

while (@@fetch_status = 0 and @KeyCount > 1)

begin

delete from r101_flow where product_id = @product_id and r101_reorg_date =
@r101_reorg_date and r101_key = @r101_key and rct_rec_num < @rct_rec_num
fetch c_r101dup into @product_id, @r101_key, @r101_reorg_date, @keyCount, @rct_rec_num
end

close c_r101dup

deallocate c_r101dup



Procedure Name: sp_RemoveDuplicate401Keys

@pid numeric(9)

AS

declare @r401_key varchar(10), @select_date smalldatetime, @product_id numeric(8,0), @r401_rec_num decimal(9), @r401_reorg_date smalldatetime, @keyCount numeric(8,0)

declare c_r401dup cursor for

select product_id, r401_key, r401_reorg_date, count(r401_key) as cnt, max(r401_rec_num) as maxRecNum
from r401_flow where product_id = @pid group by product_id, r401_key, r401_reorg_date order by
product_id, cnt desc, r401_key

open c_r401dup

fetch c_r401dup into @product_id, @r401_key, @r401_reorg_date, @keyCount, @r401_rec_num

while (@@fetch_status = 0 and @KeyCount > 1)

begin
delete from r401_flow where product_id = @product_id and r401_reorg_date =
@r401_reorg_date and r401_key = @r401_key and r401_rec_num < @r401_rec_num
fetch c_r401dup into @product_id, @r401_key, @r401_reorg_date, @keyCount, @r401_rec_num
end

close c_r401dup

deallocate c_r401dup



Procedure Name: sp_UpdateCamp_Statues

```
@product_id numeric(9),  
@camp_status_name varchar(20),  
@camp_status_descr varchar(250),  
@actionParam varchar(20),  
@kname varchar(20)
```

AS

```
If (@actionParam='AddRec')  
begin  
insert into camp_statuses (product_id, camp_status_name, camp_status_descr) values (@product_id,  
@camp_status_name, @camp_status_descr)  
end  
else  
If (@actionParam='EditRec')  
begin  
update camp_statuses set camp_status_name = @camp_status_name, camp_status_descr =  
@camp_status_descr  
where product_id = @product_id and camp_status_name = @kName  
end
```



Procedure Name: sp_UpdateCamp_Types

```
@product_id numeric(9),  
@camp_type varchar(50),  
@camp_type_descr varchar(250),  
@actionParam varchar(20),  
@kname varchar(20)
```

AS

```
If (@actionParam='AddRec')  
begin  
insert into camp_types (product_id, camp_type, camp_type_descr) values (@product_id, @camp_type,  
@camp_type_descr)  
end  
else  
If (@actionParam='EditRec')  
begin  
update camp_types set camp_type = @camp_type, camp_type_descr = @camp_type_descr  
where product_id = @product_id and camp_type = @kName  
end
```

Procedure Name: sp_UpdatePanel_SubTypes

```
@product_id numeric(9),  
@panel_subtype varchar(50),  
@panel_subtype_descr varchar(250),  
@actionParam varchar(20),  
@kname varchar(20)
```

AS

```
If (@actionParam='AddRec')  
begin  
insert into p_sub_types (product_id, panel_subtype, panel_subtype_descr) values (@product_id,  
@panel_subtype, @panel_subtype_descr)  
end  
else  
If (@actionParam='EditRec')  
begin  
update p_sub_types set panel_subtype = @panel_subtype, panel_subtype_descr = @panel_subtype_descr  
where product_id = @product_id and panel_subtype = @kName  
end
```

Procedure Name: sp_UpdatePanel_TestTypes

```
@product_id numeric(9),  
@test_type varchar(50),  
@test_type_descr varchar(250),  
@actionParam varchar(20),  
@kname varchar(20)
```

AS

```
If (@actionParam='AddRec')  
begin  
insert into p_test_types (product_id, test_type, test_type_descr) values (@product_id, @test_type,  
@test_type_descr)  
end  
else  
If (@actionParam='EditRec')  
begin  
update p_test_types set test_type = @test_type, test_type_descr = @test_type_descr  
where product_id = @product_id and test_type = @kName  
end
```



Procedure Name: sp_UpdatePanel_Types

```
@product_id numeric(9),  
@panel_type varchar(50),  
@panel_type_descr varchar(250),  
@actionParam varchar(20),  
@kname varchar(20)
```

AS

```
If (@actionParam='AddRec')
```

```
begin
```

```
insert into p_types (product_id, panel_type, panel_type_descr) values (@product_id, @panel_type,  
@panel_type_descr)
```

```
end
```

```
else
```

```
If (@actionParam='EditRec')
```

```
begin
```

```
update p_types set panel_type = @panel_type, panel_type_descr = @panel_type_descr  
where product_id = @product_id and panel_type = @kName
```

```
end
```



Procedure Name: spGetRPTList

```
@src_cat_name varchar(30),  
@src_sub_cat varchar(50)
```

AS

```
SELECT report_id,rpt_name, rpt_description, location, rpt_filters  
FROM rpt_cats  
WHERE src_cat_name = @src_cat_name  
AND src_sub_cat = @src_sub_cat
```